Interactive Spacetime Reconstruction in Computer Graphics

Von der Carl-Friedrich-Gauß-Fakultät
der Technischen Universität Carolo-Wilhelmina zu Braunschweig

zur Erlangung des Grades eines
Doktoringenieurs (Dr.-Ing.)

genehmigte Dissertation

von
Kai Nikolaus Ruhl
geboren am 05. Mai 1977
in Bielefeld

Eingereicht am:	05. Juni 2015
Disputation am:	17. Juli 2015
1. Referent:	Prof. Dr.-Ing. Marcus Magnor
2. Referent:	Prof. Dr. Peter Hall

(2015)

Abstract

High-quality dense spatial and/or temporal reconstructions and correspondence maps from camera images, be it optical flow, stereo or scene flow, are an essential prerequisite for a multitude of computer vision and graphics tasks, e.g. scene editing or view interpolation in visual media production.

Due to the ill-posed nature of the estimation problem in typical setups (i.e. limited amount of cameras, limited frame rate), automated estimation approaches are prone to erroneous correspondences and subsequent quality degradation in many non-trivial cases such as occlusions, ambiguous movements, long displacements, or low texture.

While improving estimation algorithms is one obvious possible direction, this thesis complementarily concerns itself with creating intuitive, high-level user interactions that lead to improved correspondence maps and scene reconstructions.

Where visually convincing results are essential, rendering artifacts resulting from estimation errors are usually repaired by hand with image editing tools, which is time consuming and therefore costly. My new user interactions, which integrate human scene recognition capabilities to guide a semi-automatic correspondence or scene reconstruction algorithm, save considerable effort and enable faster and more efficient production of visually convincing rendered images.

Zusammenfassung

Raumzeit-Rekonstruktion in Form von dichten räumlichen und/oder zeitlichen Korrespondenzen zwischen Kamerabildern, sei es optischer Fluss, Stereo oder Szenenfluss, ist eine wesentliche Voraussetzung für eine Vielzahl von Aufgaben in der Computergraphik, zum Beispiel zum Editieren von Szenen oder Bildinterpolation. Da sowohl die Anzahl der Kameras als auch die Bildfrequenz begrenzt sind, ist das Rekonstruktionsproblem unterbestimmt, weswegen automatisierte Schätzungen häufig fehlerhafte Korrespondenzen für nichttriviale Fälle wie Verdeckungen, mehrdeutige oder große Bewegungen, oder einheitliche Texturen enthalten; jede Bildsynthese basierend auf den partiell falschen Schätzungen muß daher Qualitätseinbußen in Kauf nehmen.

Man kann nun zum einen versuchen, die Schätzungsalgorithmen zu verbessern. Komplementär dazu kann man möglichst effiziente Interaktionsmöglichkeiten entwickeln, die die Qualität der Rekonstruktion drastisch verbessern. Dies ist das Ziel dieser Dissertation. Für visuell überzeugende Resultate müssen Bildsynthesefehler bislang manuell in einem aufwändigen Nachbearbeitungsschritt mit Hilfe von Bildbearbeitungswerkzeugen korrigiert werden. Meine neuen Benutzerinteraktionen, welche menschliches Szenenverständnis in halbautomatische Algorithmen integrieren, verringern den Nachbearbeitungsaufwand beträchtlich und ermöglichen so eine schnellere und effizientere Produktion qualitativ hochwertiger synthetisierter Bilder.

Acknowledgments

This work would not have been possible without the support of many people: First of all, I would like to thank Prof. Dr.-Ing. Marcus Magnor for inviting me to join his research group and for his advice during my time here, and Prof. Peter Hall for his advice in writing this thesis.

I would like to thank the crowd at the Institut für Computergraphik, all of whom created a wonderful atmosphere, provided advice and encouragement when needed, and lots of fun wherever possible. Special thanks go to Prof. Dr.-Ing. Martin Eisemann and Dr. Anna Hilsmann for their postdoc advice; Felix Klose for being the most likeable and witty office mate imaginable, and him and Christian Lipski for inviting me to the Virtual Video Camera project and sharing their collective wisdom with me; Dr.-Ing. Anita Sellent and Benjamin Hell for helping me with the many mathematical intricacies one encounters; Dr.-Ing. Stephan Wenger and Thomas Neumann for their enthusiasm for and fertile discussions on optimization methods and interactivity; and Matthias Überheide and Maryam Mustafa for further proofreading.

Finally, I would like to thank my family and friends for their constant encouragement, strategic advice and for living through both challenges and triumphs of the PhD path with me, most of all my wife Cordula and our little son Theo. This one is for you.

Contents

Preface

This dissertation is based on four publications: My research on interactive stereo [REM13], on interactive optical flow [RHK+12; RKLM12], and on interactive scene flow [REH+15]. Within the dissertation, these publications are presented in the common context of interactive spacetime reconstruction. The text includes material, such as figures, data, plots, and text passages, from my published work. My advisor Prof. Dr.-Ing. Marcus Magnor is a co-author on all of my publications, for which he provided ideas and advice. The individual contributions of all other authors to the works incorporated in the dissertation are clarified in the following.

The interactive stereo approach presented in [REM13] was co-written by Martin Eisemann, who also provided valuable insights, advice and discussion over the course of the project. The initial idea, development of editing operations, implementation, and evaluation are my work.

For the interactive optical flow method presented in [RHK+12], Benjamin Hell provided valuable insights into calculus of variations and norm theory; Felix Klose and Christian Lipski the correspondence

estimation and video production background that culminated in our joint Symbiz Sound music video "Who Cares" [LKRM11a], as well as valuable advice and discussion; Sören Petersen supplied CUDA assistance in implementing the TV-L1 algorithm on the GPU. The initial idea, development and implementation of editing operations, and evaluation are my work. A later enhancement to integrate approximate depth data, published in [RKLM12], was co-written by Felix Klose, and Christian Lipski contributed valuable advice and discussion. Again, the initial idea, development, implementation, and evaluation are my work.

The interactive scene flow approach presented in [REH+15] was co-written by Martin Eisemann and Anna Hilsmann, who also provided valuable insights, advice and discussion. Dennis Franke supplied OpenCL assistance in implementing the MVSF algorithm on the GPU, and Peter Eisert contributed valuable writing advice. The initial idea, development and implementation of editing operations, and evaluation are my work.

In addition to these publications, I have authored or co-authored several publications that are loosely related to this dissertation and may provide additional insight into certain aspects of the present work or a wider overview of its field of application: An exact editing approach for dense image correspondences [KRLM11]; a volumetric editing approach for astronomical nebulae [RWF+13]; a loop-consistency measure for dense image correspondences [SRM12]; a tool chain for

multi-view rendering [KLR+11]; the making of a virtual video camera film production [LKRM11a]; a stereoscopic extension from monocular footage using dense image correspondences [KRL+11]; and a multi-view video processing approach for unsynchronized cameras [LKRM11b].

I have further authored or co-authored several publications in other fields not directly related to the topic of this dissertation. These publications are listed here for completeness: A method for gas flow analysis using multiple depth sensors [BRA+11]; and a method for motion capture using multiple depth sensors [BRB+11].

All of the above publications have been peer-reviewed. A full list of my publications including book chapters and technical reports can be found on my PhD profile webpage[1].

[1] Kai Ruhl, PhD profile: `http://www.cg.cs.tu-bs.de/people/ruhl/`

Notation

In the following, matrices and vectors are written in bold typeface: $\mathbf{A}$, $\mathbf{b}$. Lowercase indicates image space: x, and uppercase indicates world space: X. Operator ∂ denotes the partial derivate in one direction and ∇ in all directions; and $\langle . \rangle$ is the scalar product.

Recorded imagery is the source for my spatiotemporal reconstructions, with camera index k, frame offsets t and resulting images I_t^k from cameras C_k. The reference or source camera is called "hero camera" in accordance with movie production naming conventions [Fai14; Fre14; Lof12; Sey12c].

Depth is generally z, and motion is divided into horizontal U, vertical V and z-motion W in 3D, or horizontal u and vertical v motion in 2D. A solution to a spatiotemporal reconstruction problem is called $\mathbf{Q}$ in its general form, with stereo solution $\mathbf{Q}_{\text{st}} = [z]$, optical flow solution $\mathbf{Q}_{\text{of}} = [u, v]^T$, and scene flow solution $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$, respectively. Discrete coordinates in the hero camera are identified as $\mathbf{x}$ and discrete or continuous coordinates projected to a camera k at time t are identified as $\mathbf{p}_t^k$.

The term "world space" without qualifier is used for the common 3D coordinate system for all cameras; the term "world space for camera C_k" for the same coordinate system centered at the location of C_k and rotated such that the positive z-axis is the viewing direction; and the term "image space" for the 2D coordinate system representing

the pixels $\mathbf{x}$ (or $\mathbf{p}$) of C_k with a value range of $[0..\{h/w\} - 1]$ for a image resolution $w \times h$. I avoid the term "camera space" because of its ambiguity and varying definitions in literature. All my coordinate systems are left-handed (LHS). The intrinsic matrix $\mathbf{K}$ and the extrinsic matrix $\mathbf{S}$ represent the world-to-image space transformation, and π is a projection function performing this transformation.

Steps in an iterative algorithm are denoted i, and quantities changing as a result of an iteration step are denoted by superscripts in parentheses, $u^{(i+1)}$.

A glossary of recurring symbols can be found on page 147.

1 Introduction

One of the major goals of computer graphics is the creation of realistic or at least plausible images that are in line with the human experience of the real world. Today's most prominent testbed is found in computer graphics supported movies which include both captured footage and rendered models. In order to enthrall viewers and uphold suspension of disbelief, the generated imagery must not only be perceptually plausible for one time instant but also temporally consistent. Models and their motions must therefore be of high quality.

Two complementary approaches work in conjunction with each other: Modeling the 3D assets manually, and reconstructing models from real-world capture. In animated movies, modeling is used exclusively, with high-quality meshes, texturing and perceptually plausible motion rigging taking considerable resources even for models that viewers do not expect to look like the real world; on the other hand, the artistic control is at its maximum. In movies based on captured footage, reconstruction is always necessary to some extent for deep compositing with other layers, and even more important when foreground objects or

subjects are to be reconstructed. The main benefit of reconstruction is the intrinsic visual realism of surfaces, textures and motions; however, artistic control is more limited because changes in rigging and lighting are not trivial.

The main problem with reconstruction is that results are rarely error-free for a number of reasons. First, all reconstruction algorithms based on recorded images rely on finding correspondences between those images, both in spatial and temporal directions. For conditions like occlusions and disocclusions, translucent surfaces, mirrors, refractions, specularities, lighting changes or moving shadows, these correspondences cannot be trivially identified. Second, recorded imagery is almost always undersampled in some direction. The number of concurrent cameras is usually limited, as is their resolution, leading to spatial undersampling; frame rate is also often limited with respect to the velocity of recorded motions, generating temporal undersampling.

Consequently, reconstruction algorithms cannot rely on dense correspondences alone and instead are usually regularized, i.e. additional constraints are included in the reconstruction process. A typical reconstruction task comprises finding a solution $\mathbf{Q}$ that is both consistent with the observation data I represented by a *data term*, and adheres to some a-priori knowledge modeled with a *regularizer*, the latter of which is most often a *smoothness term* that promotes a smooth solution whereever the data term is ambiguous. In practice, however, data and smoothness terms are often both wrong in some places, most notably

at object boundaries. Careful algorithm construction and parameter tuning or more direct user interaction is therefore required to produce convincing results.

In my dissertation, the three main branches of spatiotemporal reconstruction are considered: Stereo as representative for the more general 3D reconstruction task in space, optical flow as the dense variant of tracking in time, and scene flow as fully dense reconstruction both in space and in time. All can benefit both from better algorithms and from better user interaction.

	stereo	optical flow	scene flow
space	×		×
time		×	×
chapter	Chapter 3	Chapter 4	Chapter 5

The most substantive efforts in the last decade have concentrated on automated algorithms, fueled in part by quantitative analysis for stereo and optical flow like the Middlebury evaluation [SS02], and by steady efforts to combine the former two into scene flow [Mor12].

With strong industry demand for stereoscopic 3D movie postproduction tools, user interaction to improve stereo results have received considerable attention [SPH+11]. User interaction for optical flow, on the other hand, appears in only a few tools[1] and research approaches

[1] RE:Vision Twixtor: `http://www.revisionfx.com/products/twixtor/`

[KRLM11; RHK+12], and user interaction for scene flow is just emerging [REH+15]. The challenge in all these approaches is that complex tools require complex interactions to influence and control their behavior. My thesis contributes novel approaches to user interaction in stereo [REM13], optical flow [RHK+12] and scene flow [REH+15].

The following chapter provides an overview of the theoretical background of my work as well as an introduction to the optimization algorithms which underlie the chosen user interactions. Chapters 3 to 5 present the different spatiotemporal modalities, including relevant background, related work, algorithms, results, and discussion. An overall conclusion and outlook are given in Chapter 6.

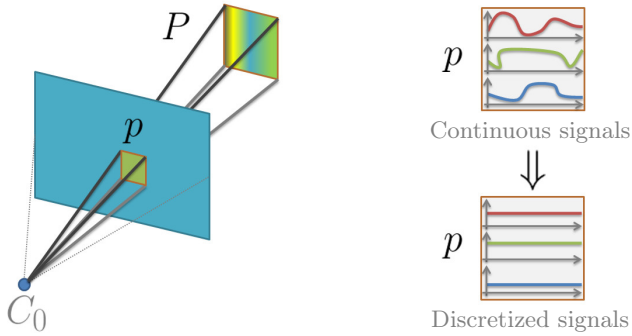
2 Prerequisites

To gain a better understanding of the user interactions in the following three chapters as well as their effects and implementation, some background knowledge may be helpful. This chapter provides that background by describing the theoretical foundations of my work.

First of all, image-based reconstruction requires a specific kind of images, whose production is described in Section 2.1. Second, to see the shortcomings of automated reconstruction approaches, an overview of spatial, temporal and spatiotemporal approaches is given in Section 2.2. Third, what constitutes *good* user interaction in the context of the outlined reconstruction issues is discussed in Section 2.3. Finally, approaches to evaluate and validate the improvements resulting from user interaction are described in Section 2.4.

2.1 Image Acquisition

The goal of image acquisition is to produce images I_t^k at time instants t for cameras C_k. These images ideally conform to a pinhole camera model, have common pixel colors for corresponding 3D surfaces, and



(a) **(b)**

Figure 2.1: Image Formation. A 2D pixel **p** is formed by projecting a 3D frustrum onto the camera sensor (a). Within this pixel, continuous RGB signals are discretized to an average RGB value, losing spatial resolution (b).

are sampled linearly such that efficient linear coordinate transforms are possible.

2.1.1 Image Formation

Consider Figure 2.1 (a) showing the idealized projection of a 3D scene surface centered at a world space point $\mathbf{P} = [X, Y, Z]^T$ onto a 2D pixel centered at a image space coordinate $\mathbf{p} = [x, y]^T$. For clarity, the sensor is shown in front of the principal point at C_0, unlike physical cameras where the sensor is mounted behind it.

In reality, the scene surface may not be planar, might not be continuous, and might in fact not be a surface at all, since the entire frustrum that is projected onto a pixel may contain partially translucent matter like fog. All these conditions create so-called "mixed pixels" which are widely considered in tomographic reconstruction [Her09] and image matting research [WC07] but largely ignored in spatiotemporal reconstruction of natural scenes.

Then, even if the surface is piece-wise planar and "Lambertian", that is, diffuse in the sense that incoming light is isotropically scattered, a pixel is composed of a RGB (red, green, blue) vector which is the per-channel integral of the observed color signals on the recorded surface, Figure 2.1 (b). This means that for regions with high color variance, high frequencies are lost to discretization.

Regarding the idealized pinhole model, a frustrum originating from one point only is not possible with physical cameras which always exhibit at least a miniscule amount of depth-of-field blur caused by the size of the camera aperture. In photography and filming, this effect is deliberately created to guide viewer focus and heighten the impression of depth. This means that the pinhole camera model is not always accurate, however since the object or subject of interest is usually in focus, resulting artifacts are often not very noticeable.

Finally, physical cameras are not noise-free, particularly for scenes with unsufficient lighting, due to signal amplification in the camera sensor; and indeed image denoising is an entire field of research [Dab10].

For the purposes of spatiotemporal reconstruction, pixel noise is treated as an outlier in the same sense as non-Lambertian surfaces that produce different RGB values in different images.

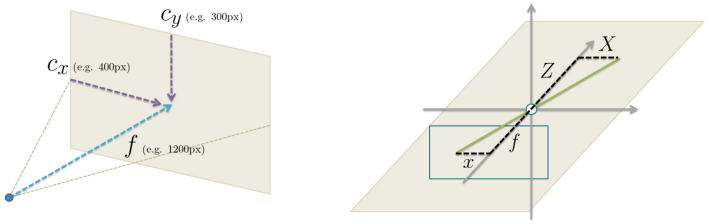
Given the above deviations from ideal image formation, purely data term-driven reconstructions are likely to exhibit artifacts for complex natural scenes, particularly given the ill-posedness of the reconstruction problem due to spatial and temporal undersampling. Consequently, scenes used for evaluation in many publications often feature mainly Lambertian scenes without translucent matter under good lighting conditions. For more complex scenes, user interaction is essential.

2.1.2 Camera Model

In the pinhole model of a camera located at $[0,0,0]^T$, with up vector $[0,1,0]^T$ and viewing direction $[0,0,1]^T$, the relation between a world space point $\mathbf{P} = [X,Y,Z]^T$ and an image space point $\mathbf{p} = [x,y]^T$ is characterized by the so-called "intrinsic matrix" $\mathbf{K}$ of the camera.

$$z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.1}$$

The parameters are shown in Figure 2.2 (a), where f is the focal length along the optical axis determining the zoom factor of the camera



(a) **(b)**

Figure 2.2: Pinhole camera model. The intrinsic camera matrix contains the focal length f and the principal point $[c_x, c_y]^T$ where the optical axis intersects the image plane (a). The X coordinate in world space is linearly related to the x coordinate in image space by $\frac{x}{f} = \frac{X}{Z}$ (b).

(sometimes divided into f_x and f_y); and c_x and c_y characterizing the principal point of the camera sensor, in order to translate from e.g. $x \in [0..640]$ to $x \in [-320..+320]$. All parameters are usually measured in pixels since calibration cannot determine the metric scale of a scene without external input [SSS06].

To illustrate coordinate conversion, consider the X component of $\mathbf{P}$, Figure 2.2 (b). Triangle equality yields $\frac{x}{f} = \frac{X}{Z}$, such that $x = \frac{f \cdot X}{Z}$ in a local image coordinate system where the center of the sensor is at $[0, 0]^T$. Adjusted for the principal point of the camera, we arrive at $x = \frac{f \cdot X}{Z} + c_x$ which is identical to the intrinsic matrix equation $z \cdot x = f \cdot X + c_x \cdot Z$, where $f_x = f$ and $z = Z$ since no scaling has

taken place. The Y component of $\mathbf{P}$ is calculated analoguously and the Z component stays the same.

Finally, when the matrix multiplication of Equation (2.1) is used, a so-called "perspective division" by z is applied afterwards to transform the 3D vector $[z \cdot x, z \cdot y, z]^T$ to the 2D image space coordinate $[x, y]^T$.

2.1.3 Calibration

When using multiple cameras, only one can be located at $[0, 0, 0]^T$ so the location and direction of the other cameras C_k has to be determined by calibration, yielding the so-called "extrinsic matrix" $\mathbf{S}$ per camera:

$$z \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{K}} \underbrace{\begin{bmatrix} r_{00} & r_{01} & r_{02} & t_x \\ r_{10} & r_{11} & r_{12} & t_y \\ r_{20} & r_{21} & r_{22} & t_z \end{bmatrix}}_{\mathbf{S}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{2.2}$$

Multiplying the homogeneous (1 added as 4th component) vector $[X, Y, Z, 1]^T$ with $\mathbf{S} = [\mathbf{R}|\mathbf{t}]$ is identical to first performing a rotation $\mathbf{R}$ on a 3D point $[X, Y, Z]^T$ and then adding a translation $\mathbf{t}$.

Calibration itself can be performed by numerous methods, one of the most established being the identification of mutually visible points in 3D space and subsequent bundle adjustment [WACS11][1], which

[1] VisualSFM: `http://ccwu.me/vsfm/`

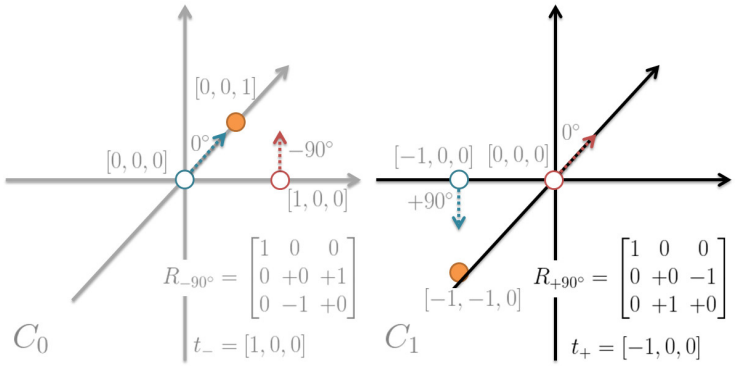


Figure 2.3: Extrinsic calibration. World space coordinates for a camera C_0 (blue) looking down the z-axis; a second camera C_1 (red) is located at $+1$ along the x-axis, with its viewing direction rotated by $-90°$ around the x-axis (a). To transform a point (orange) into the world space coordinates of camera C_1, the point is inversely rotated by $+90°$ and then shifted by -1 along the x-axis (b).

jointly estimates camera matrixes **K** and **S** as well as sparse scene geometry to minimize the overall reprojection error.

The extrinsic matrix **S** has an inverse relation to the position and viewing direction of the cameras in world space, demonstrated in Figure 2.3.

A point $\mathbf{P} = [0, 0, 1]^T$ in world space coordinates of camera C_0 is rotated by $+90°$ to $[0, -1, 0]^T$ and then translated by $[-1, 0, 0]^T$ to $[-1, -1, 0]^T$ in world space coordinates of camera C_1, Figure 2.3 (b).

Looking at Figure 2.3 (a), this is the exact inverse of the translation $[1, 0, 0]^T$ and the rotation by $-90°$ which characterizes position and viewing direction of C_1 in world space coordinates of camera C_0.

Lastly, the kinship to the so-called "SLAM" or Simultaneous Localization And Mapping problem [NLD11] in robotics should be noted: Both bundle adjustment [SSS06] and SLAM need 3D points that are visible across multiple images, and both output the location of the cameras; SLAM is usually specialized for moving robotic platforms.

2.1.4 Undistortion and Rectification

Both intrinsic and extrinsic matrix operations require a linear relationship between world and image space as described by the rectilinear projection of the pinhole camera model. However, physical lenses produce different optical aberrations, most commonly radial distortion which is often removed by remapping recorded images with radial distortion coefficients $k_1, k_2, \ldots, k_n$:

$$\text{undistort}(\mathbf{p}) = 1 + k_1||\mathbf{p}||^2 + k_2||\mathbf{p}||^4 + \ldots + k_n||\mathbf{p}||^{2n} \tag{2.3}$$

where $||.||$ is the L_2-norm and the image coordinate system is centered at $[0, 0]^T$. Figure 2.4 (a) shows an example of barrel distortion.

Commercial tools like TheFoundry Nuke™and the Matlab Calibration Toolbox as well as open source tools like the OpenCV calib3d module calculate the distortion coefficients from chessboard patterns

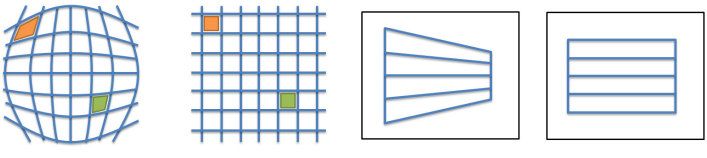


Figure 2.4: Undistortion and Rectification. Typical barrel distortion is most noticeable at the borders of the recorded image (a). Rectification requires aligning two images to each other, using perspective transformation and scaling (b).

or other line-based imagery [SAB02]. Undistortion is usually a pre-processing step because including it in the optimization objective would create a non-linear problem. In the present work, Nuke[2] has been used for removing lens distortion.

Rectification as a subfield of image registration [ENM11] is often used in stereo estimation. Essentially, a homography between two camera images is found such that the pixel rows of the two images are aligned. Both images are then resampled in a second preprocessing step, Figure 2.4 (b). This reduces the stereo problem to a 1D disparity (x-shift) search. Disparity is inversely related to the depth in world space; note that a disparity difference of 1 pixel equals different z-extents depending on the absolute z-location: The closer to the viewer, the less z-extent is covered [TSF12].

[2]TheFoundry Nuke™: `http://www.thefoundry.co.uk/nuke/`

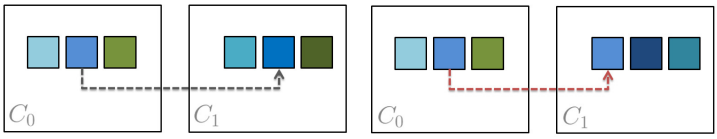


Figure 2.5: Color grading. A little darker palette in the second image still allows correct per-pixel matching of the center pixel (a). Moderate addition of blue to all pixels in C_1 leads to erroneous matching (b).

In the present work, depth is estimated along epipolar lines on unrectified images for both stereo and scene flow. Image rectification is not used in order to avoid the information loss of a second re-sampling step, and mentioned here only for completeness.

2.1.5 Color Grading

Even with identical camera models and configuration in a multi-view setup, miniscule differences in sensor sensitivity can lead to noticable color differences between camera images. Even if the same camera is used in the case of optical flow, lighting changes due to e.g. clouds for outdoor footage can produce an equivalent effect.

Since spacetime reconstruction algorithms generally rely on "color constancy", meaning same color for same surface, this can have adverse optimization effects particularly when neighboring surfaces have almost the same color.

Consider Figure 2.5, showing 3 horizontally neighboring pixels in two cameras C_0 and C_1. In correspondence estimation, shift in color is tolerable as long as neighboring pixels in C_1 are still sufficiently different to the center pixel in C_0. However, when neighboring pixels in C_1 look more like the center pixel in C_0 than the original target, the data term will optimize towards an erroneous correspondence. The same principle also applies to grayscale images.

Aligning the colors between images is known as "(color) grading" in industry and supported by tools like TheFoundry Nuke$^{\text{TM}}$ or Adobe AfterEffects$^{\text{TM}}$, while in research the term "color transfer" is used [HLKK14]. In the present work, Nuke has been used for grading.

2.2 Dense Spacetime Reconstruction

The goal of dense spacetime reconstruction is to find per-pixel correspondences $\mathbf{Q}$ between images either in an unconstrained way, such as in optical flow between two arbitrary images, or constrained by epipolar geometry as in stereo or scene flow[3].

In stereo, we strive to attain $\mathbf{Q}_{\text{st}} = [z]$ with depth z in world space of a camera C_0 in accordance with the known intrinsic and extrinsic matrixes $\mathbf{K}$ and $\mathbf{S}$ from Section 2.1.2. This is equivalent to finding a disparity d_z in image space on rectified frames, where the coordinate

[3] While scene flow originally referred to 3D motion only [VBR+99], it has since evolved to generally include depth or 3D position [Mor12].

transform has been simplified by aligning the epipolar lines onto the same scan lines (pixel rows).

In optical flow, we search for a 2D motion vector $\mathbf{Q}_{\text{of}} = [u, v]^T$ in image space that directly relates pixels in one image to the other. No constraints are given but, no depth information is conveyed either.

In scene flow, we strive to attain a 4D vector $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$ with a depth component z in world space of a camera C_0, and 3D motion components in general world space independent of any camera, with horizontal component U, vertical component V and z-component W. Depth z is constrained by $\mathbf{K}$ and $\mathbf{S}$ as in stereo; U, V, W are unconstrained as in optical flow, but this time in world space.

2.2.1 Regularized Optimization

The above variables are usually estimated by inverse methods, which are ubiquitious in computer vision and graphics [Kas92; PTK89]. Often, estimation problems are modeled as an energy function that is then minimized. A *data term* enforces the fidelity of the correspondences $\mathbf{Q}$ with respect to the input images:

$$E_{\text{data}}(\mathbf{Q}) = \int_{\Omega} \sum_{k=0}^{N_k} \sum_{t=0}^{N_t} \psi_{\text{data}}\Big(|I_0^0(\mathbf{p}) - I_t^k(\pi[\mathbf{p}, \mathbf{Q}(\mathbf{p})])|\Big) d\mathbf{p} \qquad (2.4)$$

where Ω is the domain of a reference image I_0^0 over its pixels $\mathbf{p}$, N_k the number of cameras, N_t the number of time steps, ψ_{data} some penalty

function like the Charbonnier penalty $\psi(s^2) = \sqrt{s^2 + \epsilon^2}$ [SRB10], and π the projection operator that maps points $\mathbf{p}$ from the reference image to images in other cameras and/or time steps using the correspondences $\mathbf{Q}$ and the pertinent intrinsic and extrinsic matrixes $\mathbf{K}$ and $\mathbf{S}$ where necessary. If all pixels can be mapped to the other images and the color constancy assumption is satisfied, then the energy should be minimal.

However, as outlined in Section 2.1.1, these assumptions are generally not fulfilled everywhere. While a sparse estimation could just ignore those regions, dense estimation requires the use of a regularizer enforcing some a-priori knowledge about the scene, which in spacetime reconstruction is usually a "piecewise smoothness"-assumption resulting in a *smoothness term*:

$$E_{\text{smooth}}(\mathbf{Q}) = \int_{\Omega} \psi_{\text{smooth}}\Big(|\nabla \mathbf{Q}(\mathbf{p})|\Big) d\mathbf{p} \tag{2.5}$$

with ∇ being the sum of the partial first derivates w.r.t. the variables in $\mathbf{Q}$, introducing a so-called "first order regularization"; higher-order regularizations are also possible [RGPB12] but not used in the scope of the present work. The penalizer ψ_{smooth} is often the same as ψ_{data}, but can generally be chosen freely.

The total energy then is a weighted combination of data and smoothness terms, and the task at hand is to find the $\mathbf{Q}$ that minimizes the energy E.

$$\arg\min_{Q} E(\mathbf{Q}) = E_{\text{data}}(\mathbf{Q}) + \lambda E_{\text{smooth}}(\mathbf{Q}) \tag{2.6}$$

with λ being a user-defined parameter, often in the range [0..1] for normalized terms, to enforce more or less smoothness for scenes with greater or smaller ambiguities. Note that the remaining energy contribution from only the data term is often called "costs" of a match, while the energy remaining from combined data and smoothness terms is called "residual".

There are multiple ways to solve the optimization problem, outlined in Figure 2.6. *Local* methods define a so-called "support window" around each pixel $\mathbf{p}$ for $\mathbf{Q}$ and find a solution with low cost within that support; local methods are therefore often called "cost-filtering methods" [MW15]. Since the support window will often encompass object boundaries and thereby include different objects, a major challenge is to find a heuristic that reduces the support to pixels belonging to the same object. Often, color similarity in the reference image as well as distance to the center pixel play a role. The parameter space is usually sampled once for each pixel, i.e. a user-defined search window is tested for the minimal support window cost for each $\mathbf{Q}$ candidate, yielding a so-called "cost volume". Note that the energy is local per-pixel in the

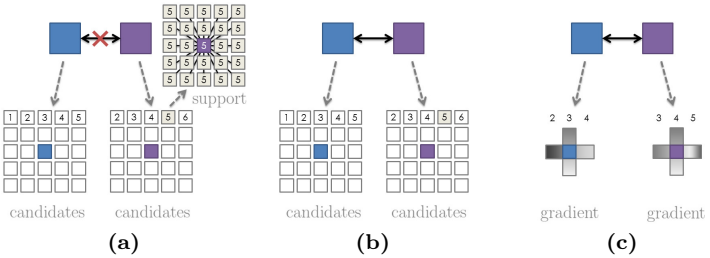


Figure 2.6: Regularized optimization in 2D. Local methods have no inter-pixel communication, but the candidates in the search window are each evaluated over a support region (a). Global loopy belief propagation tests each candidate by itself, and communicates the entire belief to its neighbors (b). Global variational methods only optimize along the local energy gradient, and communicate the results to neighbors (c).

sense that no neighbor is directly influenced; instead, support pixels are just asked for their respective per-pixel matching costs to achieve an implicit smoothness, Figure 2.6 (a). Also note that local methods can often be rewritten as energy minimization problems, contradicting the traditional view that local and global methods are intrinsically separate [MW15].

Global methods solve for each pixel individually without a support region and propagate current costs and/or solutions, often over the 4-neighborhood on the pixel grid. This allows a $\mathbf{Q}$ at one pixel $\mathbf{p}$ to potentially influence $\mathbf{Q}$ at all other pixels $\mathbf{q} \in \Omega$. Popular estimation

methods include belief propagation [FH06] and variational optimization [BBPW04], both of which are iterative methods.

Loopy belief propagation [FH06] samples the parameter space for each pixel, but unlike local estimation the costs for all candidates are sent to and received from neighboring pixels, Figure 2.6 (b). These messages are then merged with the local belief, leading to a global propagation over multiple iterations. Since samples for a pixel can be numerous even for a 2D problem ($n \times n$ samples for a search window of width and height n) and more for higher-dimensional problems, loopy belief propagation has massive memory and runtime requirements.

Variational optimization [BBPW04] shares similarities to gradient descent methods and needs comparatively less memory by allowing only one solution at any time, Figure 2.6 (c). Based on the calculus of variations, the energy E is partially differentiated w.r.t. each of the variables in $\mathbf{Q}$ and then set to zero; this is essentially the necessary optimality condition of the Euler-Lagrange equations for Equation (2.6) [BBPW04]. Since only the necessary but not the sufficient optimality condition is fulfilled, variational optimization may converge to a local minimum if Equation (2.6) is not convex [Cha04].

A plethora of other global methods exists, including graph cuts [KZ04] and tree-reweighted message passing [Kol06]. The present work uses a local cost-filtering approach [HRB+13] in Chapter 3 on stereo, and global variational optimization [BMK13; ZPB07] in Chapter 4 on optical flow and in Chapter 5 on scene flow.

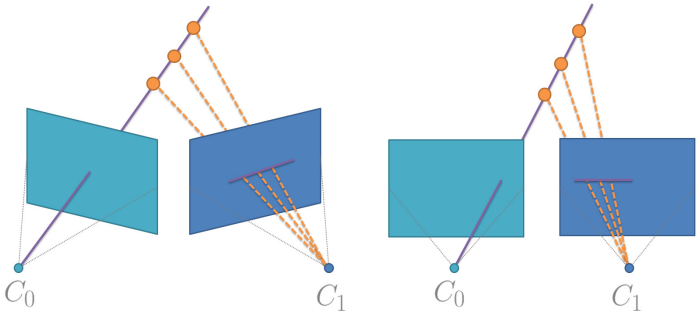


Figure 2.7: Stereo epipolar geometry. A pixel in one camera C_0 appears somewhere on the epipolar line in camera C_1 depending on its depth (a). If the camera images have been rectified, the epipolar line resides on the same scanline (pixel row) as in the source image (b).

2.2.2 Stereo

Among the correspondence estimation problems, binocular stereo can be called the most benign in the sense that searching for a 1D solution $\mathbf{Q}_{\text{st}} = [d_z]$ given two input images I_t^0 and I_t^1 from cameras C_0 and C_1, Figure 2.7, is comparatively fast and consumes little memory. This allows both local and global methods including loopy belief propagation, Figure 2.8, and also allows acceleration through GPU implementations where usually less memory is available than on the CPU.

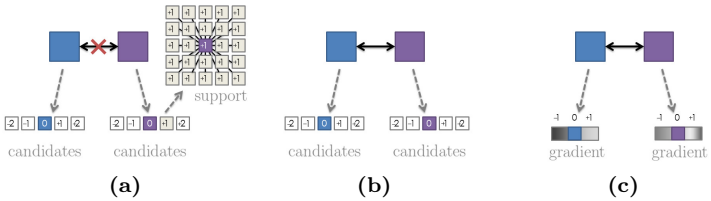


Figure 2.8: Stereo optimization in 1D. Local methods evaluate each candidate over a support window (a). Global belief propagation tests each candidate by itself, and communicates the entire belief to its neighbors (b). Global variational methods optimize along the local energy gradient (c).

A comprehensive state-of-the-art list of stereo algorithms is maintained by the Middlebury stereo evaluation[4] [SS02]. While recent publications prove their accuracy on more complex scenes such as the automotive scenes of the KITTI stereo dataset[5] [GLU12] or the rendered Sintel movie[6] [BWSB12] stereo scenes, the Middlebury index still serves as a common base benchmark.

In designing a stereo algorithm, the choice of the data term or cost penalizer ψ is the first challenge. Per-pixel matching dissimilarity $\psi(I_t^0(\mathbf{p}) - I_t^1(\mathbf{p}+d_z(\mathbf{p})))$ is often truncated in the form $\min(\tau_z, \psi(I_t^0(\mathbf{p}) - I_t^1(\mathbf{p}+d_z(\mathbf{p}))))$ with a user-defined threshold τ_z to attenuate the influence of outliers [HRB+13]. Another strategy to improve robustness to lighting changes is to perform pixel matching on gradient images

[4] Middlebury stereo evaluation: `http://vision.middlebury.edu/stereo/eval/`
[5] KITTI stereo: `http://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php`
[6] Sintel stereo: `http://sintel.is.tue.mpg.de/stereo`

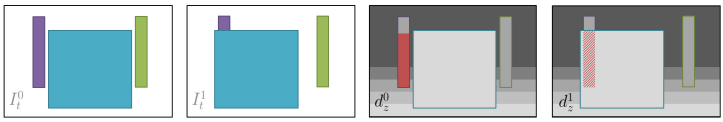


Figure 2.9: Stereo views. Left (a) and right (b) view do not always see the same objects. Consequently, depth maps for the left (c) and right (d) view exhibit unmatchable occlusion and hidden disocclusion regions, respectively.

∇I_t^0 and ∇I_t^1 instead, or in addition to color matching [MSZ+11]. Costs may also include a weighted support region either explicitly as in cost-filtering [HRB+13] and equivalently in support region aggregation [MSZ+11], or implicitly by using feature descriptors [LYT11; TSF12] which in turn describe an image region around a center pixel.

Occlusions are the biggest hurdle in stereo estimation since surfaces might only be visible in one view, Figure 2.9 (a,b), leading to unmatchable regions, d_z^0 in Figure 2.9 (c). If a local support region is used, those parts of the support that are determined to belong to the same surface, but have better color matches, will determine a plausible d_z solution; in Figure 2.9 (a,c), pixels at the bottom of the violet bar would require support pixels from the top of the bar to find the correct target. On the other hand, if a global smoothness is used, the unmatchable pixels will be determined by the smoothness term alone, leading to a linear interpolation between foreground and background z values; in Figure 2.9 (a,c), pixels at the top of the vi-

olet bar would only correctly propagate to the bottom if anisotropic smoothness [WTP+09] was used in place of the more common isotropic smoothness term. Disoccluded regions like d_z^1 in Figure 2.9 (d) do not produce errors in the depth map itself; however to correctly re-create I_t^0 from I_t^1 using d_z^1, disparities for the disoccluded regions would need to exist twice, which is not possible when using the classic one-layer depth map. Therefore, disoccluded regions can only be identified but no correct solution can be given in the standard stereo formulation.

Occluded and disoccluded regions are commonly detected using symmetry, for stereo called "left-right-consistency check". For this, d_z^0 and d_z^1 are computed for cameras C_0 and C_1, respectively; subsequently all unsymmetric entries are removed, e.g. for d_z^0 whereever $d_z^0(\mathbf{p}) \neq -d_z^1(\mathbf{p}+d_z^0(\mathbf{p}))$. The empty regions are then usually filled with background d_z values, using various methods of disparity propagation [DYLT05; SLK05; WJYG08].

While local and global approaches have long been considered separate, recent approaches combine both in a two-step fashion [MW15; SSS14], where the faster, non-oversmoothing local estimation is run first and then used as initialization and/or cost volume restriction in the slower but subpixel-precise global estimation.

Chapter 3 is based on a realtime local stereo algorithm [RHB+11] running on the GPU.

2.2.3 Optical Flow

Unlike the correspondences in stereo, which are constrained in 3D world space either explicitly by epipolar geometry or implicitly via image rectification, Figure 2.7, correspondence search for $\mathbf{Q}_{\text{of}} = [u, v]^T$ in optical flow is unconstrained in 2D since algorithms work exclusively in image space and do not deliver any 3D information.

While traditionally optical flow has been used for motion estimation between two frames I_0^k and I_1^k from the same camera C_k, frames from different cameras can also be used because different viewpoints are similar to ego-motion between two frames; however in that case, differences in color grading and noise must be taken into account. Unlike in stereo and scene flow, the cameras do not need to be synchronized since the image-space correspondence search allows for abitrary 2D motion vectors.

As in stereo, the Middlebury index[7] [BSL+11] is the base evaluation with KITTI[8], Sintel[9] and others supplying more complex scenes.

Unlike in stereo, global approaches and in particular variational methods have long dominated in optical flow research due to their automatically subpixel-precise estimation using the local energy gradient in combination with warped images in a multi-scale image pyramid [BBPW04]. Real-time capable versions using an alternating TV-L1

[7] http://vision.middlebury.edu/flow/eval/
[8] http://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php?benchmark=flow
[9] http://sintel.is.tue.mpg.de/results

optimization with a coupling term have followed [ZPB07], as have large displacement variants incorporating sparse features [BBM09]. It might also be argued that hole filling in stereo is an easier task than in optical flow since the filling direction and the notion of "background" is known in stereo but not in optical flow; this can favor the smoothness term of global optimization, particularly if anisotropic propagation is included [WTP+09], over the multi-step post-processing strategies necessary in local approaches [BYJ14].

Loopy belief propagation has had good success in optical flow estimation [LLN+12], more so than in stereo where local approaches abound, and in comparison to scene flow where propagating a 4D belief is prohibitively memory-consuming. Where variational optimization can only operate on the gradient of a scalar energy term, belief propagation can use a pixel value vector of arbitrary length including hundreds of values for feature descriptors [Low99; RRKB11; TSF12], as can local methods. However, belief propagation is very far from real-time and needs excessively many labels when subpixel accuracy is desired. It is therefore not suitable for interactive approaches.

Occlusion presents the biggest challenge in optical flow (as in stereo), being the most frequent source of data term errors, Figure 2.10. Here too, occluded pixels do not have a proper target in the other image, Figure 2.10 (c), while disoccluded pixels do not have a $[u, v]^T$ entry in the classic one-layer flow map, and would need a secondary layer, Figure 2.10 (d).

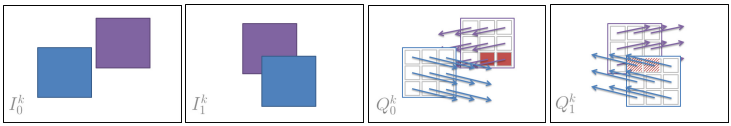


Figure 2.10: Optical flow. Motion between frames (a) and (b) can lead to (dis)occlusions. Thus, flow maps for the first (c) and second (d) frame contain unmatchable occlusion and hidden disocclusion regions, respectively.

Occlusions and disocclusions can be detected using symmetry (as in stereo), here called "forward-backward symmetry check" for optical flow [ADPS07; LLM10], or by using so-called "loop-consistency" when three or more images are considered [SRM12]. Recent approaches also model occlusion explicitly; the main challenge is that any number of layers can overlap at the same spot, creating a non-linear problem that can be solved by introducing an auxiliary occlusion map, enabling joint motion and occlusion estimation [SLP14].

Chapter 4 is based on a realtime global variational optical flow algorithm [ZPB07] running on the GPU.

2.2.4 Scene Flow

Scene flow is the joint spatial and temporal estimation of a scene[10], and the next logical step after estimating spatial stereo and temporal

[10] The original version concerned 3D motion only [VBR+99], but the contemporary definition includes depth or 3D position [Mor12].

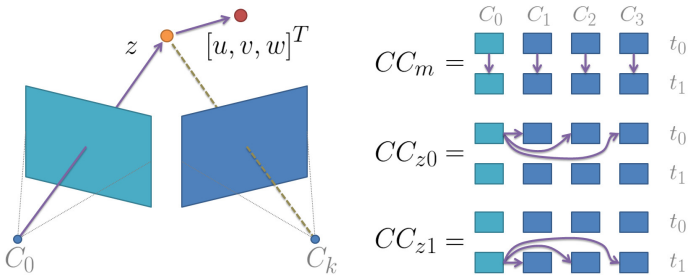


Figure 2.11: Scene flow. Both depth z from a hero camera C_0 and world space motion $[U, V, W]^T$ are estimated (a). Color constancy is tested both temporally (CC_m) over times t_0 and t_1 and spatially ($CC_{z0,z1}$) over cameras $C_{0..3}$ (b).

optical flow, where temporally consistent stereo and spatially consistent optical flow can be seen as in-between solutions [Mor12]. Often, one camera C_0 is designated the "hero camera" and depth and motion are estimated from that perspective, yielding a joint $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$ with the motion $[U, V, W]^T$ in world space, Figure 2.11 (a). The correspondence estimation is comparatively well defined due to more than two images being available, allowing color constancy checks both temporally and spatially twice at frames t_0 and t_1, Equation (2.4) and Figure 2.11 (b).

However, the joint estimation is also algorithmically more intricate, which might be the reason that fewer scene flow algorithms have been developed than stereo or optical flow algorithms [Mor12]. Additionally,

the evaluation of scene flow quality is more difficult if the error is to be measured in 3D because ground truth data of real-world scenes should be acquired by measurement means that are an order of magnitude more precise than camera measurements [NML+13], a problem also prevalent in time-of-flight (ToF) sensor research [KBKL09]. For this reason, no universally accepted standard evaluation currently exists for scene flow, and the predominant validation uses either synthetic scenes where depth and motion have been rendered in a second pass [Mor12], or reprojected stereo/flow which can then be evaluated using Middlebury, KITTI, Sintel and others as in the stereo and optical flow evaluation of the previous two subsections [Mor12].

Scene flow started as stereo and optical flow refinement [VBR+99]. First variational approaches computed stereo and optical flow jointly in image space [HD07; WBV+11]; later methods added extrinsic camera calibration [VBZ+10] for a combined 7D estimation. In contrast, 3D world space approaches model the desired $\mathbf{Q}_{\text{sf}}$ directly, Equation (2.4), but the energy formulation is more involved [BMK13; VSR11]. Belief propagation methods for scene flow exist but are restricted to very small images [IM06] due to massive memory requirements. Sparse or semi-dense scene flow variants also exist [CSH11; HB11] but as for stereo/flow do not address the matching problems of dense estimation with regard to occlusion and color constancy violations. Regarding local approaches, combined local/global approaches [QDC13] seem to be favored over purely local methods, though patch or super pixel-

based global algorithms share some similarity in the sense that a central pixel and its immediate neighborhood are tested during optimization [KLM10; VSR13], which can also be a strategy in global optical flow methods [SLP14].

Chapter 5 is based on a global variational world space scene flow algorithm [BMK13] running on the GPU.

2.3 User Interaction Efficiency

Editing dense image correspondences shares some similarities with traditional image editing in the sense that the depth of an image can be expressed as a grayscale depth map with some z-bounds $[z_{\min}..z_{\max}]$ where e.g. white represents $z_{\min}$ and black $z_{\max}$, with white/black values [0..1] in float notation and [0..255] in byte notation. Similarily, horizontal, vertical and z-motion within some motion bounds can each be represented by a grayscale image; e.g. for horizontal motion $[-u_{\max}, +u_{\max}]$, the middle gray value 0.5 denotes no motion, a black value 0.0 maximal leftward motion and a white value 1.0 maximal rightward motion. Motion can also be color-coded, which is common for visualization, but finding the correct color of a 2D motion vector for editing purposes becomes harder compared to gray scale values, even more so if a 3D scene motion is encoded in RGB channels.

2.3.1 Pixel-Precise Methods

Given grayscale depth and motion images, the full editing capabilities of established commercial tools such as Adobe Photoshop™ and AfterEffects™, TheFoundry Nuke™, and others can be used for pixel-precise improvements, which is particularly useful for fronto-parallel, rigidly moving surfaces. Also, if rotoscoping data, i.e. object segmentation outlines, is available as is often the case in visual media productions, pixel-based propagation of correct depth and motion values is further simplified.

The main issue with this method is that any relation to the recorded images used in the spatiotemporal reconstruction is completely lost, allowing arbitrarily misplaced $\mathbf{Q}$ values with high energies both in E_{data} and E_{smooth}, Equation (2.6), for perfectly valid matching regions. This lack of support by an underlying reconstruction leads to increased editing times which are only partially offset by the high degree of sophistication in image editing tools as well as the advanced education and skills of visual artist using these tools [Sey12a].

2.3.2 Approximate Methods

An area where approximate editing tools have been prevalent for a long time is image segmentation and matting [WC07]. Since selecting the outline of a subject or object in a pixel-precise manner is time-consuming and difficult, the majority of segmentation approaches

require an artist to merely place scribbles somewhere on foreground and background regions [WC07]. Using the pixels marked by the scribbles, usually two color models as well as some distance transform are determined for foreground/background, and the remaining non-marked pixels in the image are weighted and subsequently labeled either foreground or background.

Matting approaches complement segmentation by addressing mixed pixels near the segmentation outline, where a pixel contains color components from both foreground and background, Figure 2.1. In this case, it would be very difficult for an artist to determine foreground and background colors manually. Therefore, the color models of the user scribbles as well as additional color models around the segmentation outlines can be used to find the most probable color separation [LLW08].

Another well established guided editing approach is "snapping", where a comparatively approximate user-defined location is moved to the nearest image location that is the local optimum to some constraint. Unsurprisingly, snapping behaviour has also been researched in image segmentation and matting where again color models and distance transforms are used to determine a pixel-precise target outline and mixed pixel decomposition [WAC07].

Introducing these types of approximate user input as well as new editing capabilities to dense image correspondence estimation is the

subject of the present work. Chapter 3–5 present such editing methods for stereo, optical flow, and scene flow, respectively.

2.4 Visual Quality Assessment

The most prominent form of evaluation for the quality of correspondence fields $\mathbf{Q}$ is quantitative [BSL+07; SS02]. For this purpose, some ground truth data, either rendered synthetic scenes or measured real-world scenes, must be available as in the Middlebury [SS02], KITTI [GLU12], Sintel [BWSB12], and other data sets. Error measurement is performed predominantly in 2D image space, where popular metrics include average endpoint error or average angular error [BSL+07; SS02]. The main advantage of this quantitative evaluation is its objectiveness and easy application. The main drawback is that no perceptual weighting takes place: A few correspondence field errors in a salient region may cause noticable artifacts in some image rendered using $\mathbf{Q}$, while a large amount of small errors may be de-facto invisible, but at the same time contribute to a larger quantitative error.

Recognizing the need to take the human visual system into account, the field of visual quality metrics has provided numerous alternative quantitative evaluation methods [LK11] incorporating various saliency models, e.g. in the commonly used structural similarity index SSIM [WBSS04]. However, since the human visual system itself is not fully

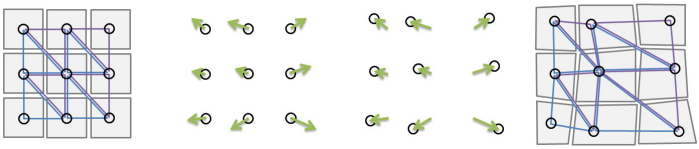


Figure 2.12: Image warping. A mesh with vertices in the center of each pixel (a) warped with $\mathbf{Q}_{\text{of}} = [u, v]^T$ (b) towards a target position (c) creates a "warped image" (d).

understood, all visual quality metrics have specific limitations which preclude universal use for visual media productions.

2.4.1 Image Warping

Qualitative comparisons of rendered to reference images make use of the human visual system of individual viewers and are therefore well suited for subjective tasks like visual media productions. They can also be used to make both artifacts and their corrections immediately obvious to a user assessing the rendered output.

Quantitative evaluation, in contrast, is often not conclusive for practical tasks particularly when an artist edits the correspondence field $\mathbf{Q}$ until the rendered output seems satisfactory, which can take varying amounts of time and effort depending on the footage as well as individual skills. A fair evaluation would therefore need to include editing effort.

In the present work, so-called "image warping" [LKRM11a] is used for the qualitative assessment of correspondence fields, Figure 2.12. For this, a mesh is constructed from triangles, with one vertex per pixel located at the pixel center, while the image data is used as texture. Note that half a pixel is lost at each image border, which is tolerable in practice. Using the correspondences $\mathbf{Q}$, the location of each pixel is shifted either in image or world space, depending on the properties of $\mathbf{Q}$, and the textured triangles are stretched or compressed accordingly.

When using a world space formulation, overlapping triangles are disambiguated using the z-order; if image space warping is applied instead, the z-order must be guessed by some heuristic [KRL+11]. Note that occlusions lead to overlapping triangles while disocclusions stretch triangles perpendicular to the occlusion edge; again, some heuristic can be used to cut overstretched triangles [LLR+10]. An ideal $\mathbf{Q}$ would then enable an exact reproduction of a target image I_t^k from a source image I_0^0:

$$I_t^k(\mathbf{p}) = I_0^0(\pi(\mathbf{Q}(\mathbf{p}))) \tag{2.7}$$

using projection matrixes $\mathbf{K}$ and $\mathbf{S}$ within the projection π to warp all pixels $\mathbf{p}$ to their proper target location. An image warped in this manner reveals all artifacts relevant to the human visual system, and only occluded, disoccluded, or unreachable regions are affected by possibly erroneous rendering.

2.4.2 Image Morphing

For visual quality assessment, one source image I_0^0 is warped fully using the correspondences $\mathbf{Q}$, i.e. by using $1.0 \cdot \mathbf{Q}$. For spatiotemporal interpolation, any other factor in the range [0..1] can be chosen with arbitrary fine steps between rendered output frames. When each of the input images I_t^k is treated as potential source image and a $\mathbf{Q}_t^k$ is estimated for each of them, warping can use multiple I_t^k as texture, with warp factors complementing each other, and the warped images can be blended using per-pixel weights to produce a higher-quality result [BBM+01], a variant of the widely-known rendering approach subsumed under the term "image morphing" [Wol98].

The following Chapters 3 to 5 all use image warping as their primary method of output image synthesis. Section 5.6 shows additional visual results using 4D image morphing.

3 Interactive Stereo Estimation

In the post production of a stereoscopic visual media production, high-quality depth or disparity maps are essential for a number of workflow components, from initial layer separation, over editing transfers, to the final depth compositing. With automatic depth estimation not being perfect as outlined in Section 2.1.1, errors lead to increased manual efforts for the artist, e.g. by requiring additional rotoscoping on the footage. Alternatively, one can fix errors in the depth maps instead of in image space since multiple subsequent workflow steps can benefit from improved depth information.

Building upon recent advances in discrete real-time stereo estimation algorithms, the approach described in this chapter guides the artist by integrating the cost volume of a stereo matching estimation into the editing parameters. The results shown in Section 3.6 further improve the good results of automated algorithms and provide an opportunity for user corrections in regions that have only partially correct estimates.

This chapter has been partially published in [REM13]. The approach is based on the cost volume stereo algorithm by Rhemann et al. [RHB+11] and the guided image filter by He et al. [HST10]. My demo video for [REM13][1] illustrates the tool's workflow.

3.1 Background

Two schools of thought compete in stereoscopic visual media production: On the one side full stereoscopic capture and editing, aided by tools like TheFoundry Ocula^{TM} [Wil09][2], and on the other side single-view capture and generation of the second view with 2D-to-3D conversion, using tools like ThePixelFarm PFDepth^{TM} [Sey12b][3]. Both approaches have in common that they require depth information.

In the conversion case, considerable effort has to be spent on rotoscoping and layer assignment, ideally resulting in temporally coherent depth maps used for second eye generation. In the case of stereoscopic capture, the data basis for automated depth map generation via stereo estimation is given, at the cost of increased capturing effort. This chapter is concerned with post-production of stereoscopic footage.

The quality of depth maps $\mathbf{Q}_{\text{st}} = [z]$ is essential since they are used for a multitude of purposes. In particular, for each editing session a camera (either left C_0 or right C_1; sometimes also a synthesized

[1] [REM13] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2013cvmp/`
[2] TheFoundry Ocula: `http://www.thefoundry.co.uk/products/ocula/`
[3] ThePixelFarm PFDepth: `http://www.thepixelfarm.co.uk/products/PFDepth`

(a) (b) (c)

Figure 3.1: Tsukuba scene (a) from Middlebury [SS02] used for fully-automated (b) vs. guided (c) depth estimation. The automated approach [RHB+11] solves Lambertian, well-textured, non-occluded areas well. Other areas benefit from approximate user guidance.

middle view) is first assigned as "hero camera". Any grading, insertion of CG and ultimately depth compositing is performed on its frames and subsequently transferred to the other camera. Both the selection of layers in the beginning, the transfer of image editing operations and the depth compositing in the end require either high-quality stereo correspondences/depth maps or, barring those, manual image corrections in all stages of post-processing. These corrections are usually performed on the captured or intermediate images, all in image space [Sey08]. Tools for image processing are therefore well developed in order to reduce artist effort [Sey08].

The complementary approach pursued in this chapter aims at fixing the depth maps $\mathbf{Q}_{\mathrm{st}}$ instead of fixing the resulting errors in image space, Figure 3.1. Keeping in mind that working on depth is not as

intuitive as working on images, only approximate user input is required in the spirit of the "snapping" behavior found in many common image editing applications. Starting with a fast initial stereo estimation method based on cost volume filtering [RHB+11], the top-performing local stereo method in 2011 on the Middlebury index [SS02], the cost volume is kept afterwards for editing purposes. Any artist operations on the depth map can now be performed in so-called "cost blocks" inside that cost volume, Figure 3.2. In particular, depth estimation can be constrained to consider only depths inside the block, and the z-resolution can be locally increased to obtain smoother results in high-salience areas, thus combining the precision of automated estimation with human scene understanding.

The presented work is only possible thanks to recent advances in near real-time, GPU-based, local stereo estimation algorithms. My main contribution is the notion of interactive depth map editing using approximate, cost volume-guided, locally refined depth estimation, useful wherever human scene recognition trumps the capabilities of stereo matching: Occlusions, noise, specularities, lighting differences and other violations of the color constancy assumption, which are common failure cases for matching as outlined in Section 2.2.

3.2 Related Work

Stereo estimation has been an active research area for the last decades and is still improving, as outlined in Section 2.2.2. Due to the large search range and sharp discontinuities of disparities, discrete methods with a fixed number of labels are more prominent than variational approaches, unlike in e.g. optical flow estimation where movements are usually comparatively smaller (though dedicated long-range optical flow algorithms do exist, e.g. [BBM09]). At the moment, discrete methods outperform continuous ones, according to the Middlebury index [SS02].

For editing operations, local optimizations which are real-time capable are needed. The recently proposed fast cost-volume filtering approach by Rhemann et al. [HRB+13; RHB+11] was the best-performing local method of 2011, and ranked 9th overall at that time; it is based on the guided image filter by He et al. [HST10] which has linear complexity due to its $O(1)$ box filter [Cro84]. I wrote an OpenCL implementation that is fast enough to ensure interactivity (e.g. 0.3s on a Tsukuba image pair at 384×288 [SS02] on a Nvidia GTX 590).

Since basically all correspondence estimation algorithms rely on well-behaved scenes, with sufficient texture and Lambertian surfaces which allow unambiguous matches, there are numerous real-world cases where human scene understanding can either support the estimation or correct its errors.

Interactive depth correction and guidance aims at providing the user with intuitive tools to aid the underlying depth estimation algorithm in otherwise difficult and ambiguous cases. In stereo conversion, a 2D video is converted to a 3D video by manually establishing a depth map for the input frames. As "automatic conversion methods are currently not sufficiently robust for general applications" [SPH+11], high quality methods are often manual, mostly relying on simple depth painting or adjusting segmented layers of the images, and as a result, are very expensive, with costs of up to $100,000 per minute of converted footage [SPH+11]. More sophisticated methods use scribble-based interfaces to draw depth and intelligently interpolate the remaining pixels [GWCO09; WLF+11] or use a set of sparse depth (in)equalities to add depth to cartoons [SSJ+10].

Given more than a single image per frame, user interaction aids stereo matching by guiding the underlying image correspondence algorithm. Typical ways are specifying sparse ground control points which serve as ground truth to estimate the depth for the remaining pixels [WY11], providing approximate correspondences which can then be refined by the underlying correspondence algorithm [KRLM11; RHK+12; RK09], or by removing outliers for better depth interpolation [CSD11].

Interestingly, user interaction is heavily used in video post-processing tools such as Ocula^{TM} by The Foundry[4] , e.g. for parallax optimization, color adjustment or detail enhancement. However, they almost always

[4]TheFoundry Ocula: `http://www.thefoundry.co.uk/products/ocula/`

create the necessary disparity maps in an automatic preprocess as the assumption is that the precision of the depth map is sufficient or given for synthetic scenes [PBH12] which is not the case for the more complex scenarios considered in this work. In contrast, my assumption is that the initial quality of the depth map is so insufficient that some means for depth map correction has to be provided.

3.3 Algorithm

My GPU depth estimation method uses a variant of the fast cost volume filtering approach [RHB+11], a discrete method which takes two color images as input. Given left and right cameras C_0 and C_1, and recorded views (images) I_t^0 and I_t^1 with discrete pixel coordinates $\mathbf{x} = [x, y]^T \in \Omega$ and RGB color values in the range [0..1], the goal is to attain for each $\mathbf{x}$ in the left camera C_0 an optimal depth $\mathbf{Q}_{\text{st}} = [z_t^0]$ with $z_t^0 \in [z_{\min}, z_{\max}]$, discretized to labels $d \in D = \{z_{\min}, .., z_{\max}\}$ from a set D.

Towards this purpose, a 3-dimensional cost volume $\mathcal{C}_t^0(x, y, d)$ is constructed for the left view I_t^0. The first two dimensions of $\mathcal{C}_t^0$ are the image size, and the third dimension is the number of depth labels. Each entry within the cost volume is initially a truncated sum of absolute differences (SAD) between single pixels of the views, using a

projection $\pi(\mathbf{x}, d)$ from left to right view based on standard epipolar geometry from calibration [SSS06] as outlined in Section 2.1.2.

$$\begin{aligned} \mathcal{C}_t^0(x,y,d) = \quad & (1-\alpha)\cdot \quad \min(\tau_1, ||I_t^0(\mathbf{x})) - I_t^1(\pi(\mathbf{x},d))||) \\ + \quad & \alpha\cdot \quad \min(\tau_2, \left\|\nabla I_t^0(\mathbf{x})) - \nabla I_t^1(\pi(\mathbf{x},d))\right\|) \end{aligned} \tag{3.1}$$

Following the parameter settings in [RHB+11], $\alpha = 0.11$ is used to favor the color term over the gradient term and $\tau_1 = 0.03$ and $\tau_2 = 0.008$ to favor only very exact matches; all larger mismatches are treated equally, and a pixel without any good matches would have the same score for all depth candidates. With the data term set, a weighted filtering on $\mathcal{C}_t^0$ is performed to arrive at a smoothed cost volume $\widehat{\mathcal{C}_t^0}$:

$$\widehat{\mathcal{C}_t^0}(x,y,d) = \sum_{\hat{\mathbf{x}} \in \mathcal{N}_r(\mathbf{x})} \mathcal{W}_{\mathbf{x},\hat{\mathbf{x}}}(I_t^0(\hat{\mathbf{x}})) \cdot \mathcal{C}_t^0(\hat{\mathbf{x}}, d) \tag{3.2}$$

The filter weights $\mathcal{W}_{\mathbf{x},\hat{\mathbf{x}}}$ depend on the guidance image I_t^0 only [HST10], similar in spirit to the anisotropic smoothness found in many variational approaches, and are computed on pairs of pixels $(\mathbf{x}, \hat{\mathbf{x}})$ on a neighborhood $\mathcal{N}_r$ within a filter radius r:

$$\mathcal{W}_{\mathbf{x},\hat{\mathbf{x}}}(I_t^0) = \frac{1}{|\mathcal{N}_r|} \sum_{b:(\mathbf{x},\hat{\mathbf{x}})} (1 + (I_t^0(\mathbf{x}) - \mu_b)^T (\Sigma_b + \epsilon U)^{-1} (I_t^0(\hat{\mathbf{x}}) - \mu_b)) \tag{3.3}$$

The mean μ_b and the covariance matrix Σ_b model the local color distribution within the filter window, and ϵU is a 3×3 diagonal matrix with very small values for numerical stabilization. I_t^0 and μ_b are 3-vectors (the color channels) and Σ_b is a 3×3 matrix of color-channel covariances.

The weights $\mathcal{W}_{\mathbf{x},\hat{\mathbf{x}}}$ are high when both pixels are on the same side of the mean for correlating color channels and reside within a highly variant region, and are low when either the two pixels have different colors or the variance in the region is small (a good gray-image explanation is a also given in [RHB+11]). Cost filtering is performed on each depth layer, but not between depth layers since there is no guide in depth direction.

Runtime is independent of the filter radius r (here, r=9..24 is used depending on image size) when using weighted box filters based on summed area tables, instead of evaluating the weights naively. My OpenCL implementation uses a tile-based sliding-window variant which works in $O(n)$ on the GPU [HBR+11].

Finally, the depth map z_t^0 is chosen by seeking the depth label with minimal cost per pixel.

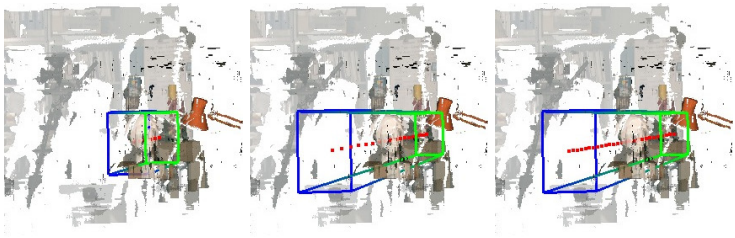
$$z_t^0(\mathbf{x}) = \arg\min_d \widehat{C_t^0}(x, y, d) \tag{3.4}$$

3.4 Problem Formulation

With the algorithm above (and others [SS02]) generally generating good initial depth estimates from stereo footage, errors cannot be avoided completely particularly when used on challenging natural scenes, requiring additional artist effort in post-production. Popular causes of artifacts include:

(**E1**) Occluded regions. Objects that are occluded differently in the two views can lose significant overlap, hindering unambiguous matching. In a typical stereo configuration, this noticeably happens for any object's left and right side, which are each only visible in one camera, Figure 2.9. The closer the object is to the camera, the more pronounced the effect becomes. Automated algorithms cannot hope to recover this error since the information is simply not available. A human user, on the other hand, is able to provide depth information for those non-visible parts by simply guessing the objects's shape.

(**E2**) Ill-textured regions. The majority of stereo algorithms for natural scenes (as opposed to controlled lab settings) rely on the color constancy assumption, which may be violated by lighting or camera sensor differences, noise, specularities, translucent objects, caustics, and others, as outlined in Section 2.1. This hinders recognition of an object in the other view. Largely uniform or repeating regions in conjunction with different occlusion boundaries in the two views (e.g. columned halls, gratings) are also not solvable with the available



(a) (b) (c)

Figure 3.2: Cost block within the initial cost volume, visualized as green-blue bounding box; z-layers are shown as red dots in the center. The cost block has a default z-extent (a) which can be increased by the user (b). The number of z-labels within the cost block can also be increased (c).

information. Again, a human user can assess which objects belong together, and thus distinguish between true and false features.

3.5 Interactive Editing

The question now is how to integrate human scene understanding in a way that minimizes interaction times, as outlined in Section 2.3. Currently, the most common way is to use image editing tools to select a region via rotoscoping or segmentation, and then use stamp, cloning and other tools to assign better depth labels [SPH+11]. This process requires pixel-precise user input. It is also completely decoupled from the cost volume, ignoring any possible guidance.

3.5.1 Cost Block Tool (BT)

The core idea of the cost block tool is to provide an approximate (and thereby fast) way for the user to specify a depth range as additional hard constraint to the cost volume. Depth values z outside this range are considered as invalid, removing a large amount of possible outliers.

The interaction starts with a lasso selection or mask in 2D image space, but instead of cloning without validation of the resulting depth, a possible range in z-direction is assigned, forming a 3-dimensional "cost block" $B_t^0(\hat{x}, \hat{y}, \hat{d}) \subseteq \mathcal{C}_t^0(x, y, d)$, Figure 3.2. In the first two dimensions, the cost block is a bounding box around the masked or selected pixels and restricts $\hat{\mathbf{x}} = [\hat{x}, \hat{y}]^T$ to come from $\Omega' \subseteq \Omega$. In the third dimension, the cost block is centered around the median depth of the selection $med(z_t^0(\hat{\mathbf{x}}))$ with $\hat{\mathbf{x}} \in \Omega'$ (other strategies are also possible) and has some extent that restricts d to come from $D' \subseteq D$. The initial extent in z-direction can either be a fixed parameter or some configurable percentile of $z_t^0(\hat{\mathbf{x}})$.

In a 3D view of the scene, Figure 3.2, both the current depth estimate and the cost block B_t^0 are visualized. An artist can now shift the cost block along the z-axis until the estimation "snaps" the depth to the most plausible position. With each step, $z_t^0(\hat{\mathbf{x}})$ is locally re-evaluated for all pixels in the mask, providing visual feedback in real-time. The z-extent of the cost block can be widened if objects in the selected area do not fit into it, or narrowed to eliminate superfluous estimates. As a

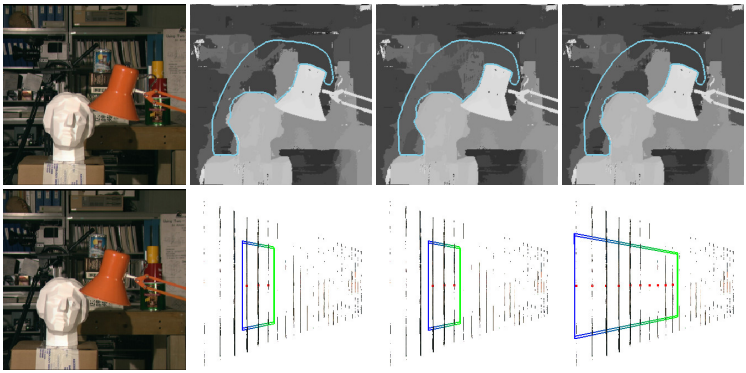
third option, the depth label subset D' can be subdivided to include more depth labels, even to the point where $|D'| > |D|$. This increases the accuracy of z-values but takes longer to compute when the cost block is large.

It should be noted that using cost blocks does not solve the problem of ill-defined regions in a mathematical sense. Instead, it merely reduces the effect of incorrect cost computation: In a narrowed set of labels $\hat{d}$, the cost block $B_t^0(\hat{x}, \hat{y}, \hat{d})$ merely evaluates to a more plausible depth $z_t^0(\hat{\mathbf{x}})$, since a search window $I_t^0(\hat{\mathbf{x}})$ has a much lower probability of being matched to a randomly low-cost window $I_t^1(\pi(\hat{x}, \hat{y}, \hat{d}))$. In the worst case, when no support information can be found within filter radius r to be aggregated into the filter weights $\mathcal{W}_{\mathbf{x},\hat{\mathbf{x}}}$, the final depth z_t^0 will be essentially random, but still within the bounds of D'. When implausible, this would require the artist to narrow down the z-extent of B_t^0 to a thin slice. Effort-wise, this would be practically equivalent to pixel-precise methods.

In essence, when thought of in the scope of the entire depth map, the user interaction cuts away large superfluous blocks from the cost volume, rather than refining the stereo matching itself.

3.6 Results

The efficiency of cost block editing is tested both on a classic Middlebury stereo scene [SS02] and on more challenging natural scenes



(a) **(b)** **(c)** **(d)**

Figure 3.3: "Tsukuba" scene from the Middlebury data set [SS02] (a). In the initial depth map [RHB+11] and scene layer representation (b), the selected area has not been properly resolved due to occlusions by bust and lamp. Within a re-evaluated cost block (c), artifacts around the bust head have vanished, but the book pile (to the left of the lamp) now has wrong depth. Increasing the z-extent of the cost block (d) corrects the book pile.

involving wider baselines and more low-textured and repeating regions. The latter are multi-view data sets from which two adjacent views are used. All examples search along epipolar lines instead of using rectified footage to avoid re-sampling and support general recording setups. Runtimes are given for an Nvidia 590 GTX graphics card, with all computation performed in OpenCL. Editing times are on the order

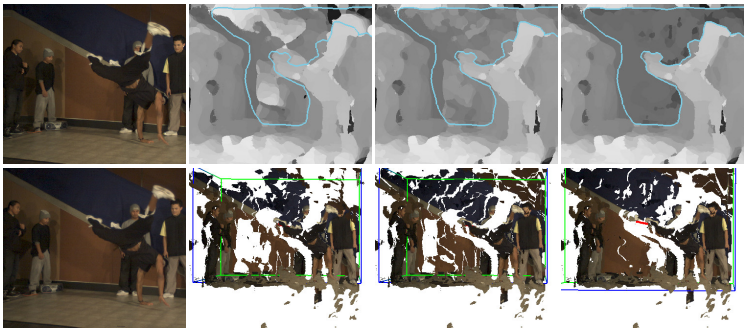
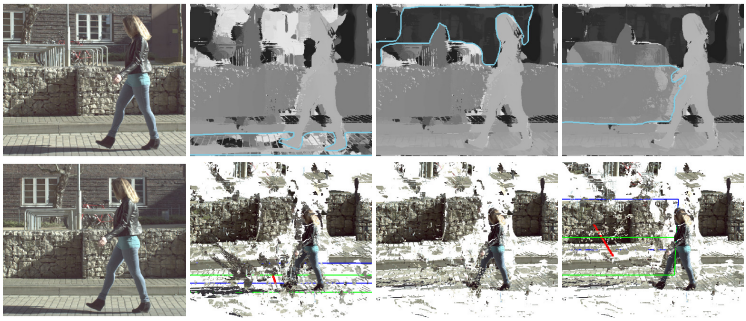


Figure 3.4: "Breakdancer" scene from Zitnick et al. [ZKU+04] (a). Due to noise and low-textured regions in the initial depth map [RHB+11] and 3D scene representation (b), large parts of the wall are ill-matched. A background region is selected (c) and evaluated with the default z positioning. The cost block is then further shifted to the back (d), which improves matching the wall.

of seconds to minutes and consist of lasso selections and cost block adjustments.

Figure 3.3 shows the "Tsukuba" scene from the Middlebury stereo evaluation data set [SS02]. Though well-textured, it contains a number of repeating regions and suffers from poor lighting. Initial depth estimation takes 0.3 seconds with 24 labels on 384×288-pixel frames. The automated cost volume filtering [RHB+11] estimates scene depth generally well, with the exception of occlusions around the bust head, lamp, and camera, and some spurious artifacts due to low lighting. An



(a) (b) (c) (d)

Figure 3.5: "Sassichan1" scene with 1 m interocular distance (a). Initial depth map [RHB+11] and 3D scene representation (b) exhibit numerous ill-matched regions due to repeating patterns, fine structures, and high number of labels. Two cost blocks repair floor and background building (c), and further cost blocks other regions (d).

area behind the lamp and bust is selected with a lasso and re-evaluated. While the occlusion artifacts vanish, the default z-extent has removed the book pile on the right table from the most probable cost volume region. Adjusting the cost block z-extent re-includes the depth region. All editing operations happen in real-time, requiring only a fraction of the 0.3 seconds needed for the entire cost volume.

Figure 3.4 shows the "Breakdancer" scene by Zitnick et al. [ZKU+04]. Initial depth estimation takes 1.9 seconds with 91 labels on 1024×768-pixel frames. The wall behind the dancers feature little texture and the recordings are noisy, leading to ambiguous matching, which is improved

by the default cost block evaluation. Shifting the wall further to the back reveals more of its true shape, namely its tilt towards the z-direction. In such cases, the number of z-labels has to be increased in order to allow for a smooth transition.

Figure 3.5 shows the "Sassichan1" scene. Initial depth estimation takes 2.2 seconds with 150 labels on 1024×540-pixel frames. The automated cost volume filtering [RHB+11] is almost to be considered a failure case due to the wide baseline coupled with many repeating, fine-structured patterns over a large number of depth labels. With a cost block constraints on the floor, artifacts are reduced, and shifting the back wall deeper removes many spurious artifacts from the front of the volume. Further coarse editing, e.g. on the bike stands, improves the result little by little. The stone wall and the back house wall could also be well approximated by a simple plane, but not e.g. the bike stand because it has some z-extent.

My demo video for [REM13][5] shows more details and a recorded editing session.

3.7 Discussion

The results show that automated stereo methods like cost volume filtering [RHB+11] produce estimates ranging from very good for well-behaved scenes such as Tsukuba to near-failure cases for complex

[5] [REM13] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2013cvmp/`

natural scenes such as Sassichan1. In all cases, shifting cost blocks around to impose constraints on the cost volume improves the result. Like the local stereo estimation that underlies the user interaction, more well-behaved scenes benefit more since the cost volume guidance is of higher quality. When the local depth estimation quality degrades too much, the guidance approach comes to its limits, and it is advisable to switch to pure image-based depth map painting [SPH+11].

Computational costs for guidance are generally low, since both the number of pixels and the number of labels is considerably less when compared to the full cost volume. Experience shows that increasing the number of labels has little influence on runtime because it is usually outweighted by the influence of the number of pixels. An interesting alternative would be to use additional variational optimization for depth refinement [KTS09] since its oversmoothing effect can be neutralized by selecting coherent regions. Furthermore, the approach currently supports only frame-by-frame edits. Integrating it into a keyframe-based framework to propagate the depth map corrections would be another way to save artist effort.

4 Interactive Optical Flow Estimation

High quality dense image correspondence estimation between two images is an essential prerequisite for view interpolation in visual media production. Due to the ill-posed nature of the problem outlined in Section 2.1.1, automated estimation approaches are prone to erroneous correspondences and subsequent quality degradation, e.g. in the presence of ambiguous movements that require human scene understanding to resolve. Where visually convincing results are essential, artifacts resulting from estimation errors must be repaired by hand with image editing tools. A new workflow alternative is investigated to fix the correspondences instead of fixing the interpolated images.

My approach combines realtime interactive correspondence display, multi-level user guidance and algorithmic subpixel precision to counteract failure cases of automated estimation algorithms. The results in Section 4.6 show that only a few interactions are sufficient to improve the visual quality considerably.

This chapter has been partly published in [RHK+12] and [RKLM12]. It is based on the realtime TV-L1 optical flow by Zach et al. [ZPB07]. My demo video for [RHK+12][1] illustrates the tool's workflow.

4.1 Background

In visual media production, view interpolation is used for a multitude of purposes, from frame upsampling (purely in the temporal domain), freeze-rotate shots (purely in the spatial domain) to combinations of both, as e.g. in our "Who Cares" video production [LKRM11a].

A typical three-stage workflow consists of estimating optical flow or dense image correspondences $\mathbf{Q}_{\text{of}} = [u, v]^T$, generating interpolated views, and correcting the interpolated frames in an image editing tool [Sey06]. The accuracy of the correspondences influences the effort one has to spend on correcting the interpolated frames; the more interpolated frames are rendered, the more favorable it is to correct an error in the correspondence map instead of in all interpolated frames.

The presented approach focuses on improving the correspondence estimation step, Figure 4.1. The proposed production workflow is to estimate dense image correspondences while integrating user interaction, generate improved interpolated views, and thus eliminate or greatly reduce the need for manual adjustment. The more difficult the cor-

[1] [RHK+12] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2012acmmm/`

(a) (b) (c)

Figure 4.1: User-guided correspondence estimation with user hints in the form of prolonged mouse clicks in green and blue (a), which have been applied in response to visual artifact detection via overlay (b). After user interaction, the overlay is much more consistent (c).

respondence estimation (e.g. fast movements, chaotic situations with much ambiguity), the greater the benefit gained by human guidance.

The steady rise in GPU power and the development of real-time capable optical flow GPU implementations [ZPB07] have made interactivity feasible also for large images. The presented approach is novel in being the first to explore interactive manipulation for dense image correspondence estimation.

4.2 Related Work

Dense image correspondence estimation and optical flow are active research areas in both computer graphics and computer vision, with impressive performance improvements in the last decade, fueled in

part by quantitative evaluation benchmarks [BSL+07] as outlined in Section 2.2.2. Contemporary algorithms achieve subpixel accuracy in continous space [WTP+09; ZPB07] or focus on large displacements by sampling [BBM09]. However, due to the ill-posed problem setting, failure cases are still frequent, e.g. in the presence of visual ambiguities, violations of brightness or gradient constancy assumptions. Occlusions are also problematic because the optical flow model simply does not consider it, although e.g. Sun et al. [SSB10] perform simultaneous layer and depth order estimation for small motions.

Flow Correction Tools aim at correcting dense correspondence fields and are a recent area of research. While supplying priors is a common technique [BN92], interactive or post-estimation correction is rare. The commercial tool Ocula™ [Wil09] edits stereo disparity maps after estimation. The commercial retime tool Twixtor™ [Sey06; Ste13a][2] provides the possibility to include mattes which influence the estimation process. The work of Klose et al. [KRLM11] focuses on post-estimation correction of image correspondences. In contrast, the present correction approach is interactive in the sense that it benefits from ongoing algorithmic refinement.

[2] RE:Vision Twixtor: `http://www.revisionfx.com/products/twixtor/`

4.3 Algorithm

My GPU correspondence estimation method uses a variant of the TV-L1 optical flow algorithm by Zach et al. [ZPB07]. Given a camera C_k producing two images I_0^k and I_1^k with discrete coordinates $\mathbf{x} = [x, y]^T$, the goal is to attain the backward flow $\mathbf{Q}_{\text{of}} = \mathbf{u} = [u, v]^T$ such that $I_1^k(\mathbf{x} + \mathbf{u}(\mathbf{x})) = I_0^k(\mathbf{x})$, where $I_{\{0,1\}}^k(\mathbf{x})$ is a grayscale brightness value and $\mathbf{u}(\mathbf{x})$ a two-dimensional image-space displacement vector.

A coarse-to-fine image pyramid with levels $L \in \{0..n\}$ is employed, 0 being the finest and n the coarsest image resolution; the number of pyramid levels depends on the image downsampling factor η, here $\eta = 0.5$ by default. Results are first computed on a coarse resolution, then upsampled to a finer resolution and refined there [BBPW04]. This allows for the assumption of small pixel motions, provided the moving objects are large enough to be identifiable on the coarser resolutions.

The TV-L1 approach [ZPB07] is an energy minimization with a coupling term that allows alternating optimizations of the data and smoothness terms, and is thereby well suited for visual analysis as outlined in Section 4.5. The overall energy to be minimized (Equation 12 in [ZPB07]) is defined as

$$E = \int_\Omega \underbrace{|\nabla \mathbf{u}|}_{\text{smooth}} + \underbrace{\frac{1}{2\theta}(\mathbf{u} - \widehat{\mathbf{u}})^2}_{\text{coupling}} + \lambda \underbrace{|\rho(\widehat{\mathbf{u}})|}_{\text{data}} d\mathbf{x} \tag{4.1}$$

Both $\mathbf{u}$ and $\widehat{\mathbf{u}}$ represent the correspondences to be estimated; the distinction is only introduced to accomodate the coupling term. The regularizer $|\nabla \mathbf{u}|$ enforces smoothness of the flow field, the residual $|\rho(\widehat{\mathbf{u}})|$ enforces adherence to the brightness constancy (data term), and λ is a weight relating data and smoothness term. The coupling term $\frac{1}{2\theta}(\mathbf{u} - \widehat{\mathbf{u}})^2$ penalizes deviations of $\mathbf{u}$ and $\widehat{\mathbf{u}}$, allowing the algorithm to perform alternate updates to $\mathbf{u}$ and $\widehat{\mathbf{u}}$ (Equation 13 and 15 in [ZPB07]). After convergence, $\mathbf{u}$ is equal or very close to $\widehat{\mathbf{u}}$.

Considering the data term in more detail, the residual ρ is defined as the difference between the warped source image I_1^k and the target image I_0^k. In order to make the function locally convex, a first order Taylor expansion is applied:

$$\begin{aligned} \rho(\mathbf{u}) &= I_1^k(\mathbf{x} + \mathbf{u}) - I_0^k(\mathbf{x}) \\ &\approx I_1^k(\mathbf{x} + \mathbf{u}_0) + \langle \nabla I_1^k(\mathbf{x}), \mathbf{u} - \mathbf{u}_0 \rangle - I_0^k(\mathbf{x}) \end{aligned} \tag{4.2}$$

For this, the flow $\mathbf{u}$ is subdivided into a fixed part $\mathbf{u}_0$ and a differentiable part $\mathbf{u} - \mathbf{u}_0$ which is optimized pointwise along ∇I_1^k. Since Taylor expansion is only valid for small distances, a coarse-to-fine warping scheme is employed where $\mathbf{u}_0$ is the upsampled flow from a coarser level.

The smoothness term $|\nabla \mathbf{u}|$ is already a convex function, so no further modification is required.

4.4 Problem Formulation

In order to formulate appropriate user actions, we first identify problems with purely automated estimation and then define user guidance operations to ameliorate them. Outside of well-behaved scenes outlined in Section 2.1, there are recurring errors that have the common trait of being readily noticable to a human observer but hard to address computationally. The most common errors include:

(**E1**) Long displacement in wrong direction or magnitude, caused by ambiguities such as several similar objects. Mismatched long displacements can also occur for small objects vanishing during the image pyramid downsampling. This is problematic only when the object motion is larger than the object size.

(**E2**) Long or short displacement in wrong direction or magnitude, caused by violation of the brightness constancy assumption, e.g. glossy objects look different in the two respective images.

(**E3**) Continuity where there should be a discontinuity, caused by uncertainty on where to split the flow. Anisotropic flow [WTP+09] is a partial remedy but cannot naturally distinguish between object boundaries and boundaries within an object, e.g. arm and wall vs. arm and sleeve: both have distinct colors but only one is an object boundary, Figure 4.1.

(**E4**) Discontinuity where there should be none. If forced to diverge, algorithms cannot make scene-based preferences about the location

of the discontinuity. Often, e.g. when shadows emerge between two time instants, regions change appearance slightly and data term-based decisions are bound to be wrong.

Problems (**E1**) and (**E2**) relate to displacement decisions, while (**E3**) and (**E4**) relate to discontinuity decisions.

4.5 Interactive Editing

To address the issues with automated estimation, my interactive user interface assists visual artists in the correspondence estimation process, Figure 4.2. The top row displays the source and target images, the bottom row shows the current flow field estimation (color-coded) together with a preview of the warped source image. This allows for fast visual quality assessment where the warped source image should ideally match the target image.

All interactive editing operations are applied while the algorithm is paused between iterations. With the user interface always showing the current estimation, the user initiates that pause upon visual identification of a mismatch, applies the guidance, and resumes into repeating the current iteration, which refines the approximate input.

In line with the objective to enhance rather than supplant existing optical flow algorithms, additional global parameter tuning can also be performed as in the automated case.

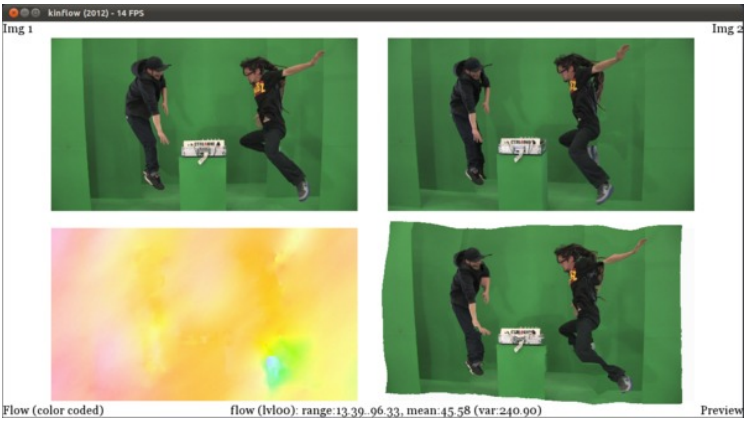


Figure 4.2: User interface with four views. Top row: Source and target image I_1^k and I_0^k. Bottom row: Correspondence estimation (color-coded) and interpolation preview (warped I_1^k). The user specifies approximate matches via prolonged mouse clicks into I_1^k and I_0^k; the matching area is determined by the duration of the click into the source image.

4.5.1 Match Tool (MT)

Algorithms with a local displacement sampling of more than one pixel [BBM09; SPC09] have tried to address the wrong direction and magnitude problem (**E1**), but ambiguities like several similar objects will still cause confusion. Meanwhile, (**E2**) remains a problem for all algorithms that rely on the brightness constancy assumption or a variant thereof, which are the vast majority.

The displacement issue is addressed with a match tool (**MT**) in the form of a user-defined offset prior $\mathbf{u}_{\text{off}}(\mathbf{x}) = [\text{offset}_u, \text{offset}_v]^T$ for a circular region of the source image I_1^k to the target image I_0^k. The use specifies the offset by first holding down the mouse button on the source image to grow a circular region at the cursor location, then clicking into the target image.

In case of a brightness constancy violation (**E2**), a sufficiently large area around the violation must be chosen. Because no solution can be found for the violated pixels, the surrounding region must enforce a common motion direction.

The user defined offset is integrated into the smoothness update minimization step, once each for horizontal and vertical motion vector index d (Equation 13 in [ZPB07]):

$$\min_{\mathbf{u}_d} \int_\Omega \left\{ |\nabla \mathbf{u}_d| + \frac{1}{2\theta} (\mathbf{u}_d - \widehat{\mathbf{u}}_d)^2 \right\} d\mathbf{x} \tag{4.3}$$

Given a $\mathbf{u}_{\text{off}}(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}^2$ at pyramid level L, the prior is integrated at the level initialization stage by replacing $\mathbf{u}_0(\mathbf{x})$ locally with $\mathbf{u}_{\text{off}}(\mathbf{x})$, creating the first $\widehat{\mathbf{u}}$. The replacement is omitted when $\mathbf{u}_0(\mathbf{x})$ and $\mathbf{u}_{\text{off}}(\mathbf{x})$ are already suffiently close (within 1 pixel distance). As a consequence, on a higher-resolution level the closeness requirement is tighter.

The user-defined motion is guaranteed to be appropriate even in ambiguous cases (**E1**) but not subpixel precise. Therefore, $\mathbf{u}_{\text{off}}(\mathbf{x})$ is

propagated to levels L to $L+m$, with m set to 10 by default, and the estimation is resumed at level $L+m$. Particularly on levels smaller than L, the optimization of $\mathbf{u}-\mathbf{u}_0$ determines the final subpixel precise placement.

The introduced offset is by nature a sharp discontinuity in the flow field, that the smoothness optimization step will try to erase. Therefore, the L1 norm $|\nabla\mathbf{u}|$ is locally replaced with the more robust Huber-L1 norm from Werlberger et al. [WTP+09] $|D^{\frac{1}{2}}\nabla\mathbf{u}|_\epsilon$ which penalizes quadratically for motions smaller than ϵ and linearly for larger motions. $D^{\frac{1}{2}}$ is a 2x2 matrix that linearly weights $\nabla\mathbf{u}_d$ with respect to the image gradient ∇I_1^k, and influences the fixed-point iteration on $\tilde{\mathbf{p}}_d$ in the update step for the smoothness term (Equation 15 in [WTP+09], Equation 10 in [ZPB07]):

$$\tilde{\mathbf{p}}_d^{(i+1)} = \frac{\tilde{\mathbf{p}}_d^{(i)} + \tau(D^{\frac{1}{2}}\nabla\mathbf{u}_d^{(i+1)} - \epsilon\tilde{\mathbf{p}}_d^{(i)})}{\max(1, |\tilde{\mathbf{p}}_d^{(i)} + \tau(D^{\frac{1}{2}}\nabla\mathbf{u}_d^{(i+1)} - \epsilon\tilde{\mathbf{p}}_d^{(i)})|)} \tag{4.4}$$

with $\tilde{\mathbf{p}}_d$ being one step in the projected gradient descent scheme [Cha04], τ being the step size, and ϵ a very small value. As in [WTP+09], $D^{\frac{1}{2}} = \exp(-\alpha|\nabla I_1^k|^\beta)\mathbf{n}\mathbf{n}^T + \mathbf{n}^\perp\mathbf{n}^{\perp^T}$ is instrumented with $\mathbf{n} = \frac{\nabla I_1^k}{|\nabla I_1^k|}$, $\mathbf{n}^\perp$ being the unit vector perpendicular to $\mathbf{n}$, and $\alpha = 3$ and $\beta = 0.5$ by default. This has the effect that the large discontinuity that has to occur due to the user-defined offset is preferably around image gradients in I_1^k, which often coincides with boundaries in user-selected

objects. Even if this is not the case, the approach will still work, but boundary regions will not be as well defined.

Violations of the brightness constancy assumption (**E2**) cannot be resolved by ρ since it is not possible to guess the "correct" color of e.g. a specularity. In this case, the region must be chosen large enough so that the smoothness term will enforce compliance to surrounding displacements.

4.5.2 Smoothness Tool (ST)

Problems (**E3**) and (**E4**) are for the most part segmentation problems as flow field discontinuities often relate to object boundaries. A layered representation requiring at least 3 input images has been addressed by Sun et al. [SSB10].

Here, the issue is addressed by a smoothness tool (**ST**), with a user-defined local data-term weight $w_{\text{data}}(\mathbf{x})$ increasing or decreasing regularization (enforcement of smoothness) on a circular area, specified as prolonged mouse clicks into I_1^k, where the click duration determines the radius of the area. Decreased regularization will allow discontinuities in the flow field, addressing (**E3**). Increased regularization will hold the region together, providing a remedy for (**E4**).

The user-defined data-term weight is integrated into the data update step (Equation 15 in [ZPB07]), again separately for each motion vector index d:

$$\min_{\mathbf{u}} \sum_{d} \frac{1}{2\theta} (\widehat{\mathbf{u}}_d - \mathbf{u}_d)^2 + \lambda |\rho(\mathbf{u}_d)| \tag{4.5}$$

Given $w_{\text{data}}(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}$ on pyramid level L, λ is locally replaced by $\lambda \cdot \exp(\eta \cdot w_{\text{data}}(\mathbf{x}))$, $\eta = 0.5$, for all regions defined in w_{data}, thereby influencing the thresholding step:

$$\widehat{\mathbf{u}} = \mathbf{u} + = \begin{cases} +\lambda\theta\nabla I_1^k & \text{if } \rho(\mathbf{u}) < -\lambda\theta|\nabla I_1^k|^2 \\ -\lambda\theta\nabla I_1^k & \text{if } \rho(\mathbf{u}) > +\lambda\theta|\nabla I_1^k|^2 \\ -\rho(\mathbf{u}) \frac{\nabla I_1^k}{|\nabla I_1^k|^2} & \text{otherwise} \end{cases} \tag{4.6}$$

The effect is that the thresholding step assumes a smaller or larger valid range $\pm\lambda\theta|\nabla I_1^k|^2$ along which to follow the image gradient according to the residual ρ.

4.5.3 Depth Tool (DT)

The match tool (**MT**) in Section 4.5.1 can also accept coarse external input, such as imprecise image correspondences from depth sensors or geometric proxies. This is particularly useful if camera ego-motion is present or if different cameras are used for the images I_0^k and I_1^k, and less so when only one static camera is used. This section has been

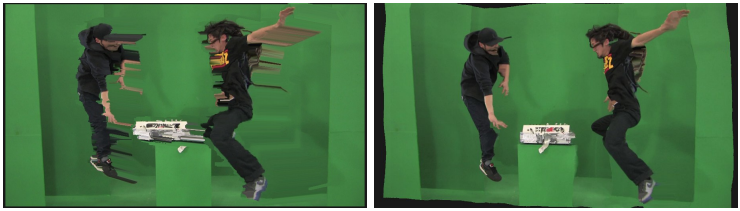
partly published in [RKLM12], and the demo video[3] illustrates the tool's workflow.

Depth sensors such as time-of-flight cameras [KBKL09] or structured light sensors like the Microsoft Kinect [SFC+11; Mic10] are inexpensive methods to capture depth information. Sensor limitations include comparatively low resolution (typically 640×480 or less) which in conjunction with high-resolution camera frames requires upsampling or other means of adaptation [KPZ+11; NIH+11; STDT08]. Limited depth range is another factor, making depth sensors most useful for foreground objects only. The vulnerability to non-Lambertian surfaces is shared with most image-based methods.

Image-space correspondences are obtained from depth sensor data by registering cameras and depth sensor using a common camera calibration as outlined in Section 2.1.3, projecting the depth points into world space [ROS10] and then reprojecting them into each of the camera views [RKLM12]. As the calibration is only accurate up to scale, the physical distance between cameras must be measured to determine the metric scale of the scene.

Building geometric proxies by hand is another method to generate depth information usable for view interpolation. While geometric proxies are often a by-product of visual media production [Sey12a], increasing the geometric detail is a time-consuming task. An alternative to manual modeling is static 3D reconstruction [SCD+06] or super-

[3] [RKLM12] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2012cvmp/`



(a) (b)

Figure 4.3: Source image I_1^k (a) warped directly by the depth-based prior $\mathbf{u}_{\text{dep}} = [u, v]^T$ and (b) warped after dense correspondence estimation guided by $\mathbf{u}_{\text{dep}}$. The large-displacement, occlusion and low-texture matching properties have been preserved while detail errors are much less present.

pixel based scene flow [KLM10], both of which can be used to create coarse geometric proxies.

Image-space correspondences can be obtained from geometry by using standard camera calibration as outlined in Section 2.1.3, and then rendering texture IDs of the modeled geometry into each of the cameras; alternatively, the depth layer can be rendered once and then reprojected into each camera as above for depth sensors [RKLM12].

Consider Figure 4.3, where the generated image-space correspondences $\mathbf{u}_{\text{dep}} = [u, v]^T$ are able to resolve occlusions, large displacements and low-textured regions particularly well; all other problems of dense correspondence estimation remain, Figure 4.3 (a). In order to remove

the errors of this first approximate solution, $\mathbf{u}_{\text{dep}}$ is used as input to the match tool (**MT**), leading to a much refined solution, Figure 4.3 (b).

Like for (**MT**), the core observation is that while the prior accuracy has high uncertainty at full image resolution $L = 0$ and the same uncertainty at lower levels, the uncertainty *when measured in pixels* decreases as L increases. Therefore, it is more favorable to include the prior at the bottom of the image pyramid $L_{\max}$ to reduce the effect of uncertain priors.

In the smoothness update step of Equation (4.3), the initial solution $\mathbf{u}_0$ is modified for each pyramid level L by performing reprojections into lower-resolution cameras C_k to build a level-dependent prior $\mathbf{u}_{\text{dep}} = [u, v]^T$, then applying this prior locally using some prior mask $\mathbf{m}_{\text{pri}} \in \{0, 1\}$:

$$\mathbf{u}_0(\mathbf{x}) = \begin{cases} \mathbf{u}_{\text{dep}}(\mathbf{x}) & \text{if } \mathbf{m}_{\text{pri}}(\mathbf{x}) \neq 0 \\ \mathbf{u}_0(\mathbf{x}) & \text{otherwise} \end{cases} \tag{4.7}$$

where $\mathbf{m}_{\text{pri}}$ is a Boolean mask requiring the prior $\mathbf{u}_{\text{dep}}(\mathbf{x})$ to exist locally, and limited either to depth pixels that are thresholded by a user-defined "foreground boundary" depth, or to depth pixels contained within some user-defined segmentation boundary $\mathbf{m}_{\text{seg}}$. One more uncertainty-reducing measure can also be used: Acknowledging that the data term $\rho(.)$ is, in good cases, more accurate than the prior $\mathbf{u}_{\text{dep}}$,

the replacement is omitted when the residual is already very low. In total, $\mathbf{m}_{\text{pri}}$ is defined as:

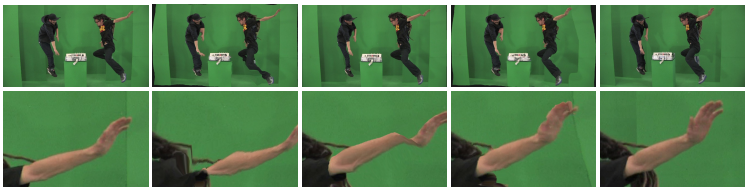
$$\begin{aligned} \mathbf{m}_{\text{pri}}(\mathbf{x}) = \mathbf{u}_{\text{dep}}(\mathbf{x}) \neq 0 \\ \wedge\ \mathbf{m}_{\text{seg}}(\mathbf{x}) \neq 0 \\ \wedge\ |\rho(\mathbf{x})| > \rho_{\text{th}} \end{aligned} \tag{4.8}$$

with ρ_{th} being a small user-defined constant, default $\rho_{\text{th}} = 0.01$. Considering Figure 4.3 again, the characteristics of the depth-based prior – correct correspondences for large displacements, occlusions and low texture – have been retained, while detail errors have been considerably reduced.

4.6 Results

Since visual plausibility is the main objective in view interpolation, a visual assessment of the warped source image I_1^k is conducted as outlined in Section 2.4, instead of using a metric such as average endpoint and angular error. The comparison algorithms are an unguided TV-L1 [ZPB07] and a dedicated large displacement optical flow estimation method [BBM09].

Figure 4.4 shows a green-screen example from a movie production [LKRM11a]. Neither TV-L1 nor LDOF (large displacement optical flow [BBM09]) are able to match arm and leg of the right actor correctly,



(a) (b) (c) (d) (e)

Figure 4.4: Jump scene from the "Who Cares" production [LKRM11a] with source image I_1^k (a) and target image I_0^k (e). Both unguided TV-L1 (b) and large displacement optical flow (c) have partial failure cases, which are remedied with mid-level user interaction and subsequent algorithmic refinement (d).

with LDOF matching the forearm well but failing for the hand. A series of six manual offset priors on level 10 allows the interactive algorithm to find an improved solution, which is then automatically refined on level 9 and upwards.

Figure 4.5 shows two frames from the Middlebury [BSL+07] "Backyard" sequence. To simulate faster movements like in real-world examples, a frame skip of 3 instead of 1 is employed. Both TV-L1 and LDOF cannot match the ball correctly, with the former simply ignoring the ball and the latter deforming it to a miniscule triangle. One user-defined offset region on level 10 solves the issue. Further problems of automated TV-L1, e.g. the green skirt, the older girl's left face side or the boy's leg, are resolved by smoothness priors or offsets.

In both examples, the interaction time was less than 30 seconds, with manual offsets as main input type.

(a) **(b)** **(c)** **(d)** **(e)**

Figure 4.5: "Backyard" scene [BSL+07] with 3 frames difference showing source image I_1^k (a) and target image I_0^k (e). Both unguided TV-L1 (b) and large displacement optical flow (c) are unable to match the ball properly. A single user operation (d) fixes the problem.

Figure 4.6 shows two frames from the Middlebury "Beanbags" sequence which contains multiple similar-looking beanbags. Again, a frame skip of 2 is used to simulate faster real-world movment. Due to the large displacement, TV-L1 cannot match the balls and does not move them, optimizing only the background. The large displacement optical flow successfully identifies the ambiguous ball movement, but mismatches fingers and parts of one ball. Multiple interactive adjustments improve the result.

Figure 4.7 shows two frames from the Middlebury "Dumptruck" sequence featuring glossy cars and thin traffic sign poles. Here, a frame skip of 3 is employed. TV-L1 mismatches the station wagon front, while LDOF creates spurious artifacts at the van door as well as erasing the traffic light pole. User guidance resolves these issues.

(a) (b) (c) (d) (e)

Figure 4.6: "Beanbags" scene [BSL+07] with 2 frames difference showing source image I_1^k (a) and target image I_0^k (e). Unguided TV-L1 (b) cannot move the balls while large displacement optical flow (c) exhibits artifacts around the hand. Several interactive user operations (d) fix the ball problem, but some artifacts on shirt and fingers remain.

In the last two examples, editing times have been under 60 seconds, again with region matching as main input type.

My demo video for [RHK+12][4] shows more details of both user interactions and the resulting improved view interpolation.

Depth Priors. Figure 4.8 shows a jump scene from the "Who Cares" music video production [LKRM11a]. The two HD cameras employed are 10 degrees and 1 meter apart, the two Kinects 20 degrees and 2 meters. The cameras and depth sensors are not subframe synchronous. Since masks are available, they are used for prior selection as in Equation (4.8), but for evaluation purposes not in the rendered frames.

[4] [RHK+12] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2012acmmm/`

(a) **(b)** **(c)** **(d)** **(e)**

Figure 4.7: "Dumptruck" scene [BSL+07] with 3 frames difference showing source image I_1^k (a) and target image I_0^k (e). Unguided TV-L1 (b) mismatches the station wagon front while large displacement optical flow (c) exhibits artifacts around the van door. Multiple interactive user operations (d) fix the problem.

Priors are integrated into the lower half of the image pyramid only, from level $\frac{L_{\max}}{2}$ until the coarsest level $L_{\max}$.

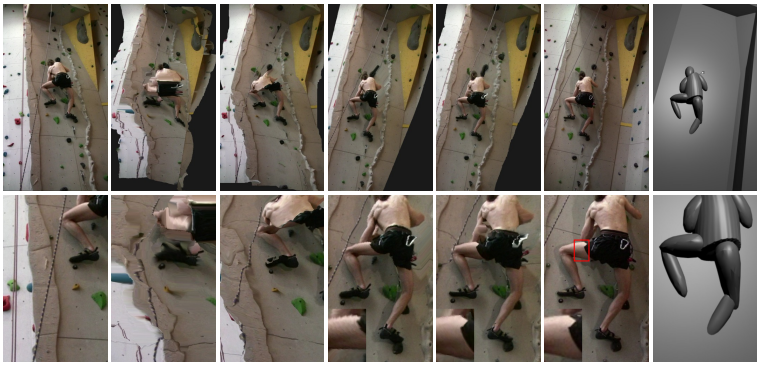
Due to the wide baseline, both unguided TV-L1 and LDOF have severe issues with the right actor, Figure 4.8 (b) and (c); particularly arm and leg, which have very large displacements compared to their size. The knee, with its black-on-black occlusion, is also hard to resolve due to lack of texture. The approximate depth guide improves the situation in all regards compared to the unguided approach, Figure 4.8 (d). Note, however, that not all details have been resolved, e.g. the right fingertips of the left actor. Still, the guided approach outperforms the unguided one in terms of visual quality also in this case.

(a) **(b)** **(c)** **(d)** **(e)**

Figure 4.8: "Who Cares" scene [LKRM11a] with source image I_1^k (a) and target image I_0^k (e). Both unguided TV-L1 (b) and LDOF (c) are unable to resolve occlusions and large displacements correctly. Approximate depth hints coupled with automated refinement (d) reduce artifacts greatly.

Geometric Proxies. Figure 4.9 shows the "Free climber" data set from Hasler et al. [HRT+09]. Two of the four hand-held HD cameras were selected, and a coarse geometric proxy was created manually to reproject a depth map into the two cameras. No masks are being used, and priors are again integrated into the lower half of the image pyramid from level $\frac{L_{\max}}{2}$ to $L_{\max}$.

Again due to the very wide baseline, the unguided TV-L1 and LDOF algorithms have severe issues with both climber and wall, Figure 4.9 (b) and (c). Applying the approximate depth-based prior directly is as good as the coarse geometric proxy but exhibits many detail errors, e.g. skinny legs, cut trousers, Figure 4.9 (d). The approximate depth guide



(a) (b) (c) (d) (e) (f) (g)

Figure 4.9: "Free climber" scene [HRT+09] with source image I_1^k (a), target image I_0^k (f) and geometric proxy (g) as user input. In unguided form, both TV-L1 (b) and LDOF (c) cannot follow the wide baseline. Direct proxy-depth based warping (d) shows good results, but details such as the left thigh (inset) are not correctly solved. Approximate depth-based priors as user input (e) eliminate many detail artifacts.

repairs many but not all of the detail errors, Figure 4.9 (e). While the improvements are not as prominent as in the previous example, the guided user input approach leads to a warped image requiring considerably reduced correction effort.

My demo video for [RKLM12][5] shows more details of depth guides as user interaction.

[5] [RKLM12] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2012cvmp/`

4.7 Discussion

Unsurprisingly, the visual quality of interpolated images improves with additional user input. The main question from a productivity perspective is whether the time spent in flow editing makes up for the time saved in final-frame editing. Compared to the pixel-precise editing approach [KRLM11] that we used in the production of the "Who Cares" music video [LKRM11a], interaction times for the presented method are short, usually a few seconds, since only approximate inputs are needed. This makes the approach valuable even if only a few output frames are interpolated.

Editing on an intermediate pyramid level is recommended as user input is in most cases imprecise; the remaining levels then refine the details of the priors globally.

The total runtime depends on the performance of the underlying optical flow and on the image size. On a Nvidia GTX590, my GPU implementation takes around 33 seconds for 720p footage, while reaching editable levels already after 5 seconds. User guidance adds a few to tens of seconds.

The general effectiveness of user input depends drastically on the edited scene. Editing rapidly changing fine structures such as hair are best left in image space, whereas object displacement can be handled well in correspondence space.

In the entire visual media production workflow, only the correspondence estimation part was considered by the presented method; rendering the output frames could also be improved, e.g. by using morphing instead of warping, as outlined in Section 2.4.2.

5 Interactive Scene Flow Estimation

High-quality stereo and optical flow maps are essential for a multitude of tasks in visual media production, e.g. virtual camera navigation, disparity adaptation or scene editing. Rather than estimating stereo and optical flow separately as in Chapters 3 to 4, scene flow is a valid alternative since it combines both spatial and temporal information and recently surpassed the former two in terms of accuracy [VSR13]. However, since automated scene flow estimation is non-accurate in a number of situations, outlined in Section 2.1, resulting rendering artifacts have to be corrected manually in each output frame, an elaborate and time-consuming task. A novel workflow is proposed to edit the scene flow itself, catching the problem at its source and yielding a more flexible instrument for further processing.

By integrating user edits into early stages of the optimization, the use of approximate scribbles instead of accurate editing is allowed, thereby reducing interaction times. The results in Section 5.6 show that editing the scene flow improves the quality of visual results considerably while requiring vastly less editing effort.

This chapter has been partly published in [REH+15]. It is based on the world-space multi-view scene flow algorithm by Basha et al. [BMK13]. My demo video for [REH+15][1] illustrates the tool's workflow.

5.1 Background

In visual media production, a number of editing operations in post-production require information about scene depth and/or motion. Due to the current popularity of stereoscopic 3D movies, depth has taken a prominent role for all manner of stereo post-production tasks [Wil09], while motion is used primarily for slow-motion, frame upsampling and motion-based effects such as motion blur.

Up to now, production tools have estimated depth and motion separately, using only two images each, which constitutes an under-determined and ill-posed problem. Since there are strong links between spatial and temporal image correspondences, i.e. object texture and boundaries are present in both, using joint optimization in the form of a scene flow $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$ is a promising direction to improve the robustness of the estimation process; stereo $\mathbf{Q}_{\text{st}} = [z]$ and optical flow $\mathbf{Q}_{\text{of}} = [u, v]^T$ are then mere projections of the scene flow $\mathbf{Q}_{\text{sf}}$ into specific cameras at specific times.

[1] [REH+15] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2015tr/`

(a) (b) (c)

Figure 5.1: Artifacts from scene flow estimation (a) have been repaired (b) by approximate scribbles (c) on the source image.

Surprisingly, for over a decade scene flow has not been able to reach the quality of dedicated stereo and optical flow methods. Only recently, scene flow algorithms have begun to outperform the former two [VSR13]. This gives rise to the idea of visual media production tools based on scene flow, Figure 5.1.

However, similar to stereo/optical flow, scene flow algorithms are not perfect, showing typical failure cases e.g. for repeating structures, occlusions and violations of the color constancy assumption, outlined in Section 2.1. Flow field artifacts manifest themselves as visual artifacts in rendered output frames, e.g. in virtual camera views. Usually, visual artists employ image editing tools such as Adobe Photoshop^{TM} to repair those visual artifacts frame-by-frame. Alternatively, they may use keyframe animation in tools such as TheFoundry Nuke^{TM} or Adobe

AfterEffects™ to model spatiotemporal transitions manually. Both are elaborate and time-consuming tasks whose effort is linear in the number of output frames or transitions.

For this reason, first editing tools for stereo and optical flow fields have recently been developed including those outlined in Chapters 3 to 4. They range from relatively direct cut&paste tools [KRLM11] to shape-fitting approaches [ZPCY13].

However, to the best of my knowledge, no such tools have been developed for scene flow yet. The presented approach is the first to provide scene flow editing capabilities, allowing interactive guidance of an ongoing scene flow estimation.

My main focus is on reducing working time. In visual media production, retouching rendered frames is often done on a massive scale within a globally coordinated, 24/7 visual production pipeline [Fai13], where pre-computation of scene flow "overnight" is already a hindrance. The presented workflow exploits the coarse-to-fine image pyramid of the optimization framework where artifacts are usually already visible on coarse to medium levels, giving the user the chance to intervene as early as possible. Once an error has been identified, approximate scribbles are used to guide the algorithm. This guidance is taken as coarse initialization or soft constraint while the ongoing optimization determines the subpixel-precise final solution. In this manner, the user benefits from both human scene recognition and subsequent algorithmic refinement.

5.2 Related Work

Scene flow as a term was arguably coined by Vedula et al. [VBR+99], describing the dense 3D geometry and motion[2] of a scene between two frames recorded by multiple cameras. If a hero camera is designated, the scene flow is represented in 4D as $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$ with per-pixel depth z and 3D motion components U, V, and W, as outlined in Section 2.2. A good survey of contemporary scene flow algorithms including both full scene flow as well as hybrids such as temporally consistent stereo is given by Mordohai et al. [Mor12]. Recent approaches include Basha et al. [BMK13] which can integrate an arbitrary number of cameras; Quiroga et al. [QDC13] which merges sparse feature correspondences with a two-camera variational formulation; and Vogel et al. [VSR13] which uses a piecewise rigid model on two cameras and shows that 2D optical flow and 1D stereo reprojected from the 4D scene flow can outperform dedicated stereo/optical flow algorithms.

Depth Editing has become popular with the latest recurrence of stereoscopic 3D movies, having two complementary approaches: stereo conversion for videos shot with a single camera, and stereo estimation for footage recorded by two cameras. In stereo conversion, 2D video is converted to 3D video by manually creating a depth map for each input frame. Current approaches use scribble-based interfaces to draw depth scribbles and interpolate the remaining pixels [GWCO09;

[2] The original definition contained 3D motion only, but has since evolved to generally include depth or 3D position [Mor12].

WLF+11], or use a set of sparse depth (in)equalities to add depth to cartoons [SSJ+10]. For footage captured with stereo cameras, user interaction is often used to guide stereo matching. Current approaches provide sparse ground truth initializations in the form of point correspondences [WY11] or matching splines [RK09] while the presented method uses matching regions; remove outliers for better depth interpolation [CSD11] or restrict the cost volume and enhance local depth resolution [REM13] as outlined in Chapter 3; use geometric model fitting and discontinuity brushes in a belief propagation framework [ZPCY13] where the presented method uses discontinuity scribbles; or modify local weights in a variational energy functional [DCSK14], which is also part of the presented approach.

Optical Flow Editing is useful for all manner of temporal effects or when employing multiple non-synchronized cameras [LLR+10]. Current approaches use cut&paste on a flow field to match regions via perspective transformation and to re-compute optical flow locally [KRLM11], or provide approximate correspondence regions which are then refined by further optimization [RHK+12] as outlined in Chapter 4, which is also part of the presented method.

Scene Flow Editing has, to the best of my knowledge, not been previously explored, presumably due to increased algorithmic complexity which leads to increased runtimes, thereby hampering interactivity. In contrast, real-time algorithms exist for stereo and optical flow, such as those used in Chapters 3 and 4, enabling interactive applications.

The closest in spirit to the presented method are the stereo editing approach by Doron et al. [DCSK14] and my optical flow editing approach from Chapter 4 [RHK+12] in the sense that both modify a variational energy functional, and the latter exploits the image pyramid for refining approximate input.

5.3 Algorithm

My GPU scene flow estimation approach is based on the well-known multi-view scene flow by Basha et al. [BMK13]. Scenes are recorded with an arbitrary number K of cameras $C_{0..K-1}$, one of which is designated as the hero camera, at two frames t_0 and t_1, making it usable not only for stereo cameras but also e.g. for trifocal cameras [Ste13b] or multi-camera rigs [BMK13].

Consider Figure 5.2 (a). For the hero camera (C_0 in the following), the goal is to reconstruct $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$ with per-pixel depth z and 3D motion $\mathbf{m} = [U, V, W]^T$ in world space, directly estimating the 3D unknowns. Assuming brightness constancy, a variational energy minimization is employed [BMK13]:

$$E(z, \mathbf{m}) = \int_\Omega \underbrace{(BC_m + BC_{z0} + BC_{z1})}_{\text{data term}} + \alpha \underbrace{(S_m + \mu S_z)}_{\text{smoothness term}} \, dxdy \quad (5.1)$$

with α balancing data vs. smoothness and μ balancing motion smoothness vs. depth smoothness. Using recorded frames I_t^k from cameras C_k at time steps $t = 0..1$, the subterms are, Figure 5.2 (b):

$$BC_m(z, \mathbf{m}) = \sum_{k=0}^{K-1} o_m^k \cdot \psi(|I_0^k(\mathbf{p}_0^k) - I_1^k(\mathbf{p}_1^k)|^2) \tag{5.2}$$

$$BC_{z0}(z) = \sum_{k=1}^{K-1} o_{z0}^k \cdot \psi(|I_0^0(\mathbf{p}_0^0) - I_0^k(\mathbf{p}_0^k)|^2) \tag{5.3}$$

$$BC_{z1}(z, \mathbf{m}) = \sum_{k=1}^{K-1} o_{z1}^k \cdot \psi(|I_1^0(\mathbf{p}_1^0) - I_1^k(\mathbf{p}_1^k)|^2) \tag{5.4}$$

$$S_m(\mathbf{m}) = \psi(|\nabla u|^2 + |\nabla v|^2 + |\nabla w|^2) \tag{5.5}$$

$$S_z(z) = \psi(|\nabla z|^2) \tag{5.6}$$

with $BC_{m,z0,z1}$ being the brightness constancy data terms for motion, depth and depth-after motion, and $S_{m,z}$ being the smoothness terms for motion and depth; occlusion maps $o_{m,z0,z1}$ deactivate the data term locally, image space points $\mathbf{p}_{t=0}^k(z)$ and $\mathbf{p}_{t=1}^k(z, \mathbf{m})$ are reprojected from world space points $\mathbf{P}_0$ and $\mathbf{P}_1$ into a camera k, and the non-quadratic robust Charbonnier penalty is $\psi(s^2) = \sqrt{s^2 + \epsilon^2}$ [SRB10]. Details of the optimization process can be found in [BMK13].

5.4 Problem Formulation

Now where does this model fail? My analysis identified six typical situations, exemplified in Figure 5.3 by both synthetic and real-world

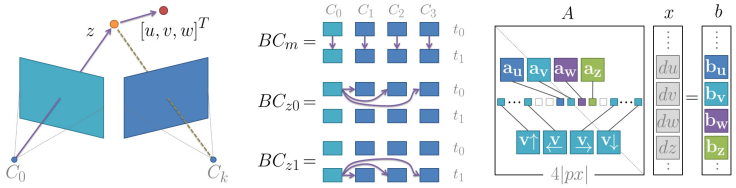


Figure 5.2: Optimization details. (a) The 4D flow $\mathbf{Q}_{\text{sf}} = [z, U, V, W]^T$ is modeled in world space, allowing for an arbitrary number of cameras to support the hero camera C_0; arrows denote depth and motion. (b) The data term comprises a motion term $BC_m(z, \mathbf{m})$, a depth term $BC_{z0}(z)$ and a depth-after-motion term $BC_{z1}(z, \mathbf{m})$; arrows denote the frames used for data term evaluation. (c) The $\mathbf{Ax} = \mathbf{b}$ system of linear equations links U, V, W and z and retrieves support information from the 4-neighborhood, here: a row for V.

scenes. Row 1 shows input frames leading to artifacts; row 2 additionally motivates the scenarios. Four problems arise from the data term **(D1–4)** and two from the smoothness term **(S1–2)**:

(**D1**) Violations of brightness or color constancy assumption, as outlined in Section 2.1. Any reflectance, specularity or other non-Lambertian property such as shadows can cause pixels in one image to simply not look the same in other images, invalidating the data terms $BC_{z0,z1}$ spatially and/or BC_m temporally. Thus, a user should be able to deactivate the data term locally, leaving the region to the smoothness term.

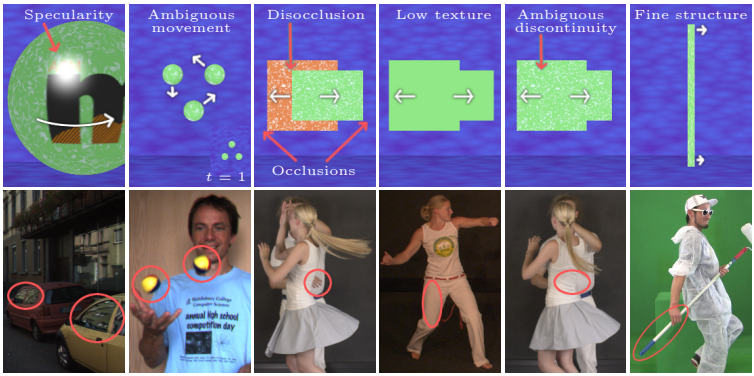


Figure 5.3: Common artifact causes. Top row: input frames leading to artifacts. Bottom row: Similar situations in real-world footage. (**D1**) color constancy violation: the ball rotates to the right while the specularity stays in place. (**D2**) ambiguous displacements of three spheres (**D3**) disoccluded region without proper source region (**D4**) low-textured regions (**S1**) ambiguous discontinuities: not all image gradients are discontinuities. (**S2**) fine-scale objects. See Figures 5.4 to 5.9 for artifact details.

(**D2**) Spatial ambiguities or large displacements. Repeating or similar structures have multiple local minima and can be incorrectly matched by the energy minimization. Also, since variational approaches handle large displacements on coarse image pyramid levels, an object smaller than its motion between t_0 and t_1 might be oversmoothed by the image pyramid during downsampling, effectively

vanishing. In these cases, an appropriate action would be to give the algorithm some correspondence hint.

(**D3**) Occluded regions. Between two images I_t^k, occluded pixels have no proper target in $BC_{m,z0,z1}$, violating the color constancy assumption. Other images might still contain the same region, so in this case a highly selective version of data term deactivation is desirable. Additionally, should no image contain the occluded region, smoothness will now fill the region from all sides, which is usually not the desired effect, so some capability to influence the smoothness propagation direction would be helpful.

(**D4**) Low-textured regions. Being a uniform case of ambiguous matching, low texture is ideally resolved by the smoothness terms $S_{m,z}$, since the data term has equal penalties everywhere. However, noise has a comparatively large influence in $BC_{m,z0,z1}$. Thus, an appropriate countermeasure either provides some hint regarding matching regions, or promotes uniform z and $\mathbf{m}$ for that region.

(**S1**) Discontinuities. The smoothness terms $S_{m,z}$ do not actively detect object boundaries, and uniformly demand e.g. $|\nabla U|$ in Equation (5.5) to be low for all pixels alike. Robust approaches avoid over-penalization of discontinuities when compared to quadratic terms but still require smoothness everywhere. Furthermore, Vogel at al. [VSR13] argue that W motion around discontinuities can often be "simulated" by U and V motion to avoid a smoothness penalty, circumventing correct discontinuity formation. Heuristics like anisotropic regulariza-

tion [WTP+09] explicitly encourage the formation of discontinuities along image gradients but cannot differentiate between true object boundaries and texture gradients (e.g. striped shirt). Thus, a user should be able to specify "true" discontinuities manually, either as sharp boundaries or with broad brushes indicating boundary candidates.

(**S2**) Inappropriate smoothness weight. Weighting the smoothness term vs. the data term is usually done globally, here with α and μ. Depending on the image content, applying different weights to different regions may be more appropriate. Setting one global weight too high results in oversmoothing, preventing deformations and glueing objects together. Setting the weight too low results in non-smooth objects distorted by ambiguities in the data term. Therefore, it is necessary to control the smoothness weights α and μ per image region.

5.5 Interactive Editing

The above analysis is distilled into four interactive editing tools that can be used in conjunction with each other: An edge tool (**ET**) in Section 5.5.1, an occlusion tool (**OT**) in Section 5.5.2, a smoothness tool (**ST**) in Section 5.5.3, and a match tool (**MT**) in Section 5.5.4. Porting the scene flow by Basha et al. [BMK13] to OpenCL yields a speedup factor of 3–5 and enables interactive feedback to the user.

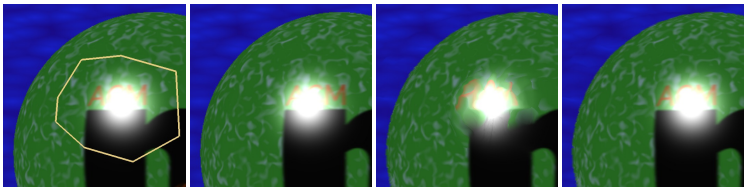
The workflow consists of observing the scene flow estimation proceed from image pyramid levels $L = L_{\max}..0$, where 0 is the finest resolution

level; at each level the hero camera image I_0^0 is warped towards all other images I_t^k and displayed to the user for visual comparison to the recorded frames so that mismatches become apparent at early levels.

To integrate the tools into the scene flow estimation, the optimization algorithm (detailed in Section 2.3 of [BMK13]) needs to be considered. The algorithm couples multi-resolution levels with two nested fixed-point iterations where outer iterations update z and $\mathbf{m}$ and re-warp the images accordingly, while inner iterations compute small increments dz and $\mathbf{dm} = (dU, dV, dW)$. At each inner fixed-point iteration, the Euler-Lagrange equations for the variables z, U, V and W are solved by constructing a system of linear equations $\mathbf{Ax} = \mathbf{b}$, Figure 5.2 (c), and solving it. $\mathbf{A}$ is a sparse quadratic matrix and has four times the number of pixels rows and columns, i.e. $4 \cdot w \cdot h$ rows and columns for images sized $w \times h$. Matrix elements close to the diagonal reference the other variables (e.g. V references U, W and z at the same pixel) and receive support information from the 4-neighborhood.

5.5.1 Edge Tool (ET)

Human scene understanding can easily distinguish true discontinuities from in-object gradients (e.g. recognizing a striped shirt as such), so the user is allowed to define undirected edge scribbles on object discontinuities in the hero camera image I_0^0, Figures 5.6 to 5.9. Pixel neighborhood across such a scribble will be ignored in the smoothness

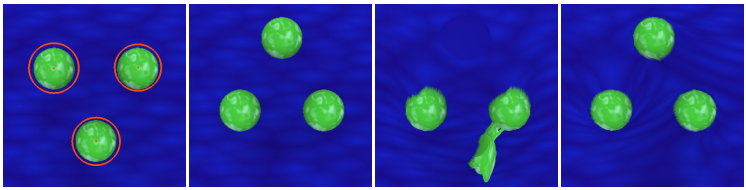


(a) UI at t_0 **(b) ref. at t_1** **(c) w/o UI** **(d) w/ UI**

Figure 5.4: Correcting a brightness constancy violation (**D1**). The specular region is marked as "occluded", deactivating the data term (a). Warped images should look like the reference (b). Automated scene flow produces a distorted "ACM" logo around the specularity (c). User interaction preserves the sphere's shape (d).

term, addressing problems (**S1**) and (**D3**). This unfortunately requires precise user input. Ideally, a broad stroke could be used to define a region in which anisotropic filtering would find the exact edge location. In practice, however, failure cases occur mostly around less visible discontinuities, Figure 5.3 (S1). Therefore, using exact scribbles is the most sensible choice.

For integration into the scene flow, the energy functional cannot be directly modified since α is applied omnidirectionally, while here a discontinuity perpendicular to the user defined edge is desired. Instead, consider the realization of S_m, Equation (5.5), into the neighborhood coefficients $v_{\leftarrow}$, $v_{\rightarrow}$, $v_{\uparrow}$ and $v_{\downarrow}$ in Figure 5.2 (c). For a pixel $\mathbf{p} =$



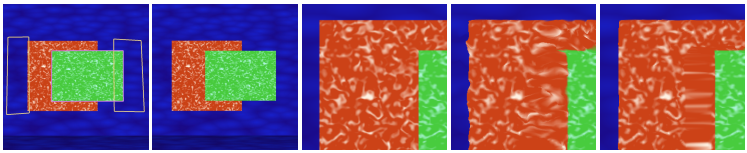
(a) UI at t_0 **(b) ref. at t_1** **(c) w/o UI** **(d) w/ UI**

Figure 5.5: Correcting ambiguous matching (**D2**). Circular matches provide a local initialization (a). Compared to the reference (b), automated scene flow cannot determine which ball belongs to which (c). The user's local initialization is enough to converge to the correct result (d).

$[x, y]^T$, e.g. $v_{\leftarrow}$ is defined as (derived from Equation 35 in Appendix C of [BMK13]):

$$v_{\leftarrow} = -\alpha \cdot \mu_v \cdot \frac{1}{2}(\text{div}^{uvw}(x, y) + \text{div}^{uvw}(x - 1, y)) \tag{5.7}$$

with div^{uvw} the divergence coefficients for U, V and W derived from S_m (div^z is used analoguously for realizing S_z, Equation (5.6), into $z_{\leftarrow}$); $v_{\leftarrow}$ is the coefficient responsible for optimizing the "left" direction of S_m. In order to deactivate the neighborhood relation, it is tested whether a segment of the edge scribble intersects the line between the center points of the current pixel and the left neighbor pixel, and if so, $v_{\leftarrow}$ is set to zero or alternatively to a user-defined small value to produce less sharply pronounced discontinuities.



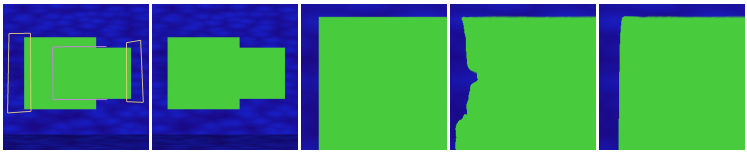
(a) UI at t_0 **(b) ref. at t_1** **(c) zoom t_1** **(d) w/o UI** **(e) w/ UI**

Figure 5.6: Correcting occluded regions to the left and right, and a disoccluded region in the middle (**D3**), all under good texturing conditions. Occluded regions are marked as such, and an edge allows the formation of a discontinuity (a). Warped images should look like the reference at t_1 (b, c). Automated scene flow produces uneven results around the discontinuity (d), whereas the corrected version pulls the flow field apart very evenly (e).

5.5.2 Occlusion Tool (OT)

Within a region defined by a closed scribble on the source image I_0^0, the data term may be deactivated w.r.t. a number of user-defined images I_t^k, Figures 5.4 to 5.9, making this tool useful both for true occlusions as well as for color constancy violations, addressing (**D1**) and (**D3**). As long as other cameras can see the region, the scribble can be defined very approximately. For a complete data term deactivation, the region can also be approximately defined but expected smoothness propagation must be considered; in practice, an additional edge can be used to stop unwanted propagation directions.

For integration into the scene flow, the occlusion variables $o_{m,z0,z1}$ in Equations (5.2) and (5.4) are modified. Consider the realization of



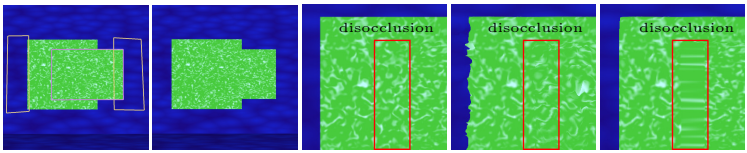
(a) UI at t_0 **(b) ref. at t_1** **(c) zoom t_1** **(d) w/o UI** **(e) w/ UI**

Figure 5.7: Correcting low-textured regions (**D4**). Same input as in Figure 5.6, but the edge can be placed almost arbitrarily (a). Compared to the reference (b, c) and to Figure 5.6, automated scene flow suffers from uneven smoothness propagation (d) due to spatially varying data term validity. The edge scribble allows for a uniform smoothness propagation towards the left (e).

BC_m and BC_{z1} into the diagonal coefficient a_v in Figure 5.2 (c). For a pixel $\mathbf{p}$, a_v is defined as (derived from Equation 35 in Appendix C of [BMK13]):

$$a_v = \alpha \cdot \mu_v \sum_{\mathbf{q}\in\mathcal{N}(\mathbf{p})} \frac{1}{2}(\mathrm{div}^{uvw}(\mathbf{p}) + \mathrm{div}^{uvw}(\mathbf{q})) \tag{5.8}$$
$$+ \sum_{k=0}^{K-1} o_m^k \cdot \Psi_m^k \cdot (\mathrm{Iw}_{t[v]}^k)^2 + \sum_{k=1}^{K-1} o_{z1}^k \cdot \Psi_{z1}^k \cdot (\mathrm{Iw}_{t[v]}^k - \mathrm{Iw}_{t[v]}^0)^2$$

with $\mathcal{N}$ the 4-neighborhood; Ψ_m and Ψ_{z1} derived from $BC_{m,z1}$; $\mathrm{Iw}_{t[v]}^k$ the relevant images (c.f. Figure 5.2 (b)) warped with the current $z/\mathbf{m}$ solution and differentiated w.r.t. v; and $\mathbf{p}$ only noted where necessary to improve readability.



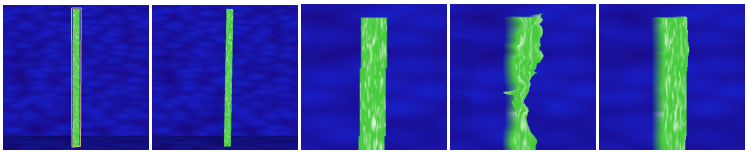
(a) UI at t_0 **(b) ref. at t_1** **(c) zoom t_1** **(d) w/o UI** **(e) w/ UI**

Figure 5.8: Correcting an ambiguous discontinuity (**S1**). As in Figure 5.6, occluded regions are marked, and the edge allows formation of the discontinuity (a). Warped images should look like the reference at t_1 except for the disocclusion region (red box) (b, c). Unaided scene flow results in craggedness at the discontinuity (d), while the corrected discontinuity is very even (e).

The occlusion variables o_m and o_{z1} are from Equations (5.2) to (5.4) (o_{z0} is used for calculating a_z). o_{z0} is locally replaced for spatial occlusions and o_m and o_{z1} for temporal occlusions, or all for total data term deactivations, e.g. for a moving specular region. When set to zero, the data terms $BC_{m,z0,z1}$ are effectively omitted and do not factor into a_v at all, leaving the smoothness $S_{m,z}$ as the only influence.

5.5.3 Smoothness Tool (ST)

Within a user-defined closed scribble on the source image I_0^0, stronger or weaker smoothness weights α and μ can be assigned, Figure 5.9, addressing (**D4**) and (**S2**). Undersmoothing can be easily solved by selecting a region and increasing α and/or μ. Oversmoothing can be solved either by decreasing α/μ or by providing an edge scribble.



(a) UI at t_0 **(b) ref. at t_1** **(c) zoom t_1** **(d) w/o UI** **(e) w/ UI**

Figure 5.9: Preserving fine structures (**S2**). The stick receives an increased smoothness weight (a). Compared to the reference (b, c), automated scene flow often smooths the background flow over the stick (d). The corrected version preserves the stick structure (e).

In practice, a common strategy is to define regions with increased smoothness as slightly too large and then using the edge tool to encourage discontinuity formation. Note that it is theoretically possible to specify smoothness and edge weights such that the tools cancel each other out, but this is not a practical problem.

For integration into the scene flow, consider a_v in Equation (5.8) and $v_{\leftarrow}$ in Equation (5.7) again. To modify the smoothness, α and μ are locally replaced by custom user values defined within the closed smoothness scribble. It is also possible to define α_Z, α_u, α_v and α_w separately (same for μ), but in practice this is rarely needed.

5.5.4 Match Tool (MT)

Starting with a closed scribble or circular region in the source image, an approximate target displacement into another image may be set

with a defined translation, rotation, and scale, Figure 5.5, similar to [KRLM11] and addressing (**D2**) as well as (**D1**) and (**D4**). For spatial matches at the same time, a guide along the epipolar line is provided to the user. For temporal matches, no guides are possible since movement is not constrained in 3D space. The match does not need to be very precise since it is used merely as initialization that is further refined by the data term in the ongoing optimization. In practice, matches are often necessary for large displacements, and subsequently incur a strong smoothness penalty S_m, Equation (5.5); this can be ameliorated by an additional edge scribble. It is also a good strategy to define matches on a coarse pyramid level L as early as possible to allow the data term to refine the match on finer levels.

For integration into the scene flow, each pixel $\mathbf{p}_s$ inside the source region is first related to a target pixel $\mathbf{p}_t$, Figure 5.10, using an affine transform based on the user-defined translation, rotation and scale. In the case of motion, both source and target pixel are projected into world space using the current z solution, yielding world space points $\mathbf{P}_s$ and $\mathbf{P}_t$, Figure 5.10 (a); the difference between them is the new motion vector $\mathbf{m}$. In the 1D constrained case of depth, camera rays $\mathbf{R}_s$ and $\mathbf{R}_t$ are projected into world space, Figure 5.10 (b); calculating the shortest line between the (possibly skewed) rays via closed form solution, the z component in the middle of the line is taken.

At each outer fixed-point iteration, the current z and/or $\mathbf{m}$ solution is locally set to the calculated value. The replacement is repeated up to a

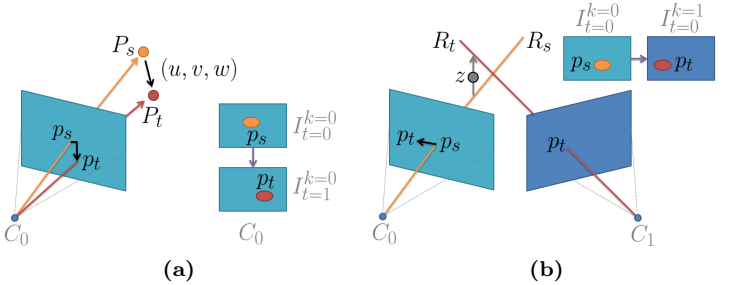


Figure 5.10: Match tool (**MT**) details. For temporal matches (a), two user-defined image locations at times t_0 and t_1 are projected into world space, and the difference is the new motion vector. For spatial matches (b), two user-defined image locations in cameras C_0 and C_k are projected as rays into world space, and the middle of the closest line between the two rays is the new depth value.

level L_d defined by the user during matching; this leads to a stable and predictable influence on the surrounding region similar to the strategy in [RHK+12] outlined in Chapter 4 (however, no anisotropic term is used since it requires regions with good object/background image gradients, which are not always given). The lower (finer) L_d, the less iterations will be performed to refine the match. Depending on the situation, this allows the user to specify either refined approximate or enforced precise matches.

5.5.5 Direct Matching

In addition to the intra-optimization tools above, this tool also allows the user to match regions after the estimation has finished, effectively emulating [KRLM11] for scene flows. However, this is only needed for rare intractable cases (e.g. two grids moving against each other), requires very precise user input, and was not used for the results in Section 5.6. It also requires waiting for the final scene flow, whereas in the presented approach, an artist can usually let the last refinement levels run unattended.

5.5.6 View Propagation Tools

When scene flows for multiple hero cameras are desired (e.g. to produce the morphed frames in the results in Section 5.6), the first solution is to reproject the world space scene flow into the other cameras. However, pixels disoccluded in the target camera will be undefined, necessitating a new scene flow calculuation.

Therefore, my implementation allows the propagation of scribbles towards other cameras and time steps according to the estimated scene flow. Smoothness and matching regions are reprojected using z and $\mathbf{m}$ obtained at the centroid of the region. Edges must either partially enclose some region or be defined in pairs that, when connected, form a closed region whose centroid can be taken. Occlusion regions are often on the background, near a discontinuity; they must either be

redefined or can be transferred by taking the centroid of a nearby smoothing or matching scribble.

In all cases, further translation/rotation of propagated scribbles is supported. For spatially propagated scribbles, usually only a minority must be corrected and even fewer redefined. Temporally propagated scribbles suffer from missing temporal symmetry more often and thus need redefinition more frequently. Compared to manual redefinition of all scribbles, the propagation approach saved considerable time in the creation of the results in Section 5.6.

5.6 Results

The interactive scene flow editing results are presented in two parts. The first part shows the hero camera frame at t_0 *warped* towards t_1, as outlined in Section 2.4.1. This approach is best suited for quality assessment because artifacts in the scene flow are directly identifiable. The second part presents results *morphed* from all cameras at both t_0 and t_1 towards a virtual camera position k_{vrt} and a time t_{vrt}, as outlined in Section 2.4.2, i.e. all images are warped towards the same virtual spatiotemporal location and then blended using per-pixel weights. This method produces the visually most pleasing results. The chapter concludes with a quantitative evaluation of the rendered output and a summative user study.

Warps for Quality Assessment. For a visual assessment of flow field quality, the hero camera's t_0 frame is shown warped fully towards t_1. By comparing the warped frame to the reference frame at t_1, flow field artifacts become directly visible. For warping, a fully connected mesh with one vertex in the center of each pixel is used as outlined in Section 2.4.1; additionally, nearest neighbor interpolation is employed because single pixels can be discerned that way. To clearly demonstrate cause and effect, 6 synthetic examples are presented with minimal user interaction, each addressing one of the failure cases identified in Section 5.4. Often, an additional edge scribble could improve the result further but would reduce clarity. Additionally, 3 real-world examples are shown where all tools have been applied in combination.

Synthetic Examples. Figure 5.4 shows a rotating textured sphere with a large specularity, addressing (**D1**). The color constancy violation is treated by specifying a full spatiotemporal occlusion (**OT**) that deactivates all brightness constancy terms $BC_{m,z0,z1}$, Equations (5.2) to (5.4). Since the scene flow model does not separate light transport and object surface, moving the specularity with the object is the geometrically correct choice and the desired effect; alternatively, preserving the specular location would be possible with an edge scribble (**ET**) to separate it from the surrounding motion. The corrected sphere shows intact letters where the automated version shows distorted ones.

Figure 5.5 shows three diffuse balls with the same texture, all moving to new locations, addressing (**D2**). The ambiguous matching problem is mitigated by providing three manual matches (**MT**) which are then refined automatically. Note that as no further edge scribbles have been applied to the example, the background is partly dragged with the balls. The corrected balls move to the correct location where the automated version shows severe misplacement.

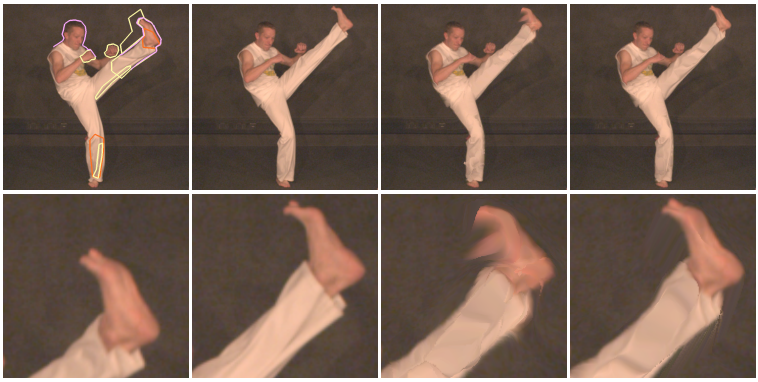
Figure 5.6 shows two textured boxes horizontally moving apart from each other, leading to occlusions of the background and a disocclusion of the red box, addressing (**D3**). The occlusion problem is solved by applying the temporal occlusion tool (**OT**) on the background. The scribbles may well encompass part of the foreground as long as the marked foreground is spatially visible in other cameras. An additional edge scribble (**ET**) around the green box targets the disocclusion problem and leads to a consistent discontinuity formation at the purple line. Note that had anisotropic smoothness been used, wavy edges would have occured as in Figure 5.6 (d) due to irregular brightness coincidences e.g. between box and background. The corrected boxes show straight borders and discontinuities where the automated version shows distorted ones.

Figure 5.7 shows the same two moving boxes from Figure 5.6 but this time untextured, addressing (**D4**). While it is impossible to determine the scene flow quality visually in the middle of the image, the same two temporal occlusion scribbles (**OT**) and one edge scribble

(**ET**) at some plausible box boundary solve the problem. Additional tests confirm that the edge scribble is indeed necessary; without it, irregular smoothness propagation from the top and bottom of the boxes arrives as incoherent horizontal U motion at the left and right rectangle borders. The corrected box preserves the straight border where the automated version produces irregular curvature.

Figure 5.8 is almost identical to Figure 5.6 but this time the two boxes have the same texture and are much more difficult to disambiguate, addressing (**S1**). The same user interaction from Figure 5.6 solves the issue here as well since it allows flow field divergence at the correct location (**ET**) and disallows impossible pixel matches in the regions occluded at t_1 (**OT**). The corrected boxes preserve straight borders and discontinuity where the automated version shows distorted results.

Figure 5.9 shows a thin, slowly moving structure that is prone to being overridden by the surrounding background due to its relatively small influence in the smoothness term evaluation, addressing (**S2**). The problem is solved by demanding a large smoothness weight for the stick, using a region scribble around the structure (**ST**). Note that an additional edge scribble would reduce the impact of brightness coincidences between stick and background texture further. The corrected stick retains its shape while the automated version deforms its shape considerably.



(a) t_0 **(b)** t_1 **(c) w/o UI** **(d) w/ UI**

Figure 5.11: Warped images. "Capoeira" scene with large displacements, low texture and shadows. The foot is matched, shadows and creases marked as occluded, smoothness increased (a). Compared to the reference at t_1 (b), the foot motion leads to streaking artifacts (c). The corrected version forms the foot better (d) given the large displacement from (a).

Real-World Examples. All four tools are used in combination on real-world footage which was recorded with 4 RED Scarlet-X at 4K resolution and 15cm interocular baseline, only approximately color-graded, and downsampled to 540p to reduce noise.

Figure 5.11 shows a Capoeira scene with fast motion and low-textured clothing featuring crease deformations and shadows. Consider the high-kicking leg. Foot and lower leg receive increased smoothness (**ST**) and an edge (**ET**) to allow large motions against a static

(a) t_0 **(b)** t_1 **(c) w/o UI** **(d) w/ UI**

Figure 5.12: Warped images. "Dancing" scene with creases in low textured clothing. A large crease is matched, the outer side marked occluded, smoothness increased (a). Compared to the reference (b), the skirt is not estimated very well (c). The corrected version resolves the motion plausibly (d), with some of the left border from t_0 (a) remaining.

background. The background above the leg is marked as temporally occluded (**OT**). An additional match (**MT**) around the ankle is required to overcome an incorrect local minimum. A clothing crease on the thigh not visible at t_1 is also marked as occluded (**OT**). Further edits include the standing leg with match and smoothing, the dark hair being edge-protected from the equally dark background, and hands being smoothed. The corrected foot shows a consistent shape where the automated version shows severe streaking artifacts.

Figure 5.12 shows a pair-dance scene with actress/actor occlusions under same-colored clothing, a classic failure case for anisotropic smoothness, and deep clothing creases. Consider the left side of the skirt. The shadowed and deforming crease is matched (**MT**) and left and right side marked as occluded (**OT**) for true temporal occlusion and color constancy violation, respectively. Further edits include smoothing the flowing hair and edge-protecting it against the background; edges around the female dancer's upper arm; increased smoothness and temporal occlusion at the male dancer's hand; and smoothing and matching around the foot and ankle. The corrected skirt shows a plausible shape where the automated version shows unrealistic folding.

Figure 5.13 shows a complex outdoor family scene with sharp depth discontinuities as well as fine structures with large motion which are usually lost in the downsampling of the image pyramid. Consider the rightmost arm, its color similarity to the right background, and the hand's large displacement relative to its size. The arm is matched (**MT**), smoothness increased (**ST**), and edge-protected (**ET**). Due to the texture similarity of arm and background, data term refinement after matching can still produce artifacts, which are suppressed with the occlusion tool (**OT**). Note that the smearing artifacts in the disoccluded region on the left side of the arm are caused by the fully connected mesh used in the warping approach. Further edits include smoothness and edge-protection around the heads; and match, small edge and data term deactivation via occlusion to repair the leftmost

(a) t_0 **(b)** t_1 **(c) w/o UI** **(d) w/ UI**

Figure 5.13: Warped images. "Family" scene with fine structures under complex background. The arm is matched, smoothed, and marked as occluded where color constancy violations remain (a). Compared to the reference (b), the arm is completely destroyed in (c). The corrected version preserves the shape of the arm (d) and stretches the disocclusion stemming from (a) evenly.

hand. The corrected arm preserves its shape where the uncorrected version tears the arm apart.

In all examples, editing times are on the order of minutes; applying the scribbles is a matter of seconds, and observing the effect forming in the ongoing optimization takes tens of seconds over several re-warp iterations. While my implementation already runs on the GPU to achieve interactive feedback, an even faster scene flow algorithm or faster GPUs would reduce total editing time further.

(a) w/o UI **(b) w/ UI** **(c) w/o UI** **(d) w/ UI**

Figure 5.14: Morphed images from 8 views (4 cameras, 2 time steps). "Capoeira" scene. Different artifacts in the views produce a halo streaking effect around the foot (c), while the corrected version features only minor motion blur (d).

Morphs for Rendering. This part shows how the presented tools can be used to improve image-based rendering quality, morphing 8 views (4 cameras at 2 time steps) based on Lumigraph rendering [BBM+01]. The virtual camera is defined by a virtual camera position $k_{\text{vrt}} = 0..3$ and virtual time $t_{\text{vrt}} = 0..1$. Frames are blended with linear temporal weighting and spatial weighting depending on the viewing angle per pixel (details in [BBM+01]). The best virtual spatiotemporal position to observe artifacts is in the middle between two cameras and times, i.e., $t_{\text{vrt}} = 0.5$ and $k_{\text{vrt}} = 0.5$, 1.5 or 2.5 respectively, where input from the 4 adjacent views are maximally warped before being blended. Below, results are shown for $t_{\text{vrt}} = 0.5$ and $k_{\text{vrt}} = 0.5$.

Figure 5.14 shows the improved visual quality of morphs with user interaction compared to morphs without user interaction, mirroring

(a) w/o UI **(b) w/ UI** **(c) w/o UI** **(d) w/ UI**

Figure 5.15: Morphed images from 8 views (4 cameras, 2 time steps). "Dancers" scene. The blended skirt artifact (c) has vanished in the repaired version (d).

the improvements of warped results, Figure 5.11. On the Capoeirista's high foot, the 4 involved views all have different artifacts, each of which are 25% visible as a halo artifact when blended. The corrected version is spatiotemporally consistent and therefore able to provide high-quality blending.

Figure 5.15 again shows that user interaction, here on the skirt, solves the shortcomings of automated estimation. The left side of the Dancer's skirt blends 4 different artifacts in the unaided case, which are replaced with a consistent appearance in the corrected version.

Figure 5.16 shows the most extreme example. Due to widely differing arm motions in the 4 automated estimates, the arm effectively vanishes during blending. In contrast, the corrected version leaves the arm entirely intact. Note that the smearing artifacts in the disoccluded region left of the arm are rendering artifacts due to warping with a fully connected mesh. In future work, a more sophisticated rendering

(a) w/o UI **(b) w/ UI** **(c) w/o UI** **(d) w/ UI**

Figure 5.16: Morphed images from 8 views (4 cameras, 2 time steps). "Family" scene. Widely differing failure modes make the blended arm almost invisible (c), while the correct version preserves the arm's shape.

removing these regions in the relevant views could improve morphing results. My demo video for [REH+15][3] shows all sequences in motion as well as the user interactions applied to improve the renderings.

Quantitative Evaluation. In addition to the visual quality, the numerical quality of the corrected scene flows compared to uncorrected ones is assessed based on the structural similarity index method (SSIM) [WBSS04], where fully-warped images are compared to the reference image at t_1, Table 5.1. The scores range from $-1.. + 1$ for dissimilar to similar. Evaluated on full frames, scores are nearly undistinguishable in all cases since artifacts are relatively small, so the presented scores are calculated on the partial images shown in Figures 5.11 to 5.13, bottom row.

[3] [REH+15] video: `http://www.cg.cs.tu-bs.de/publications/ruhl2015tr/`

Dataset	auto	user	RRE	Figure
Capoeira (foot)	0.937	0.944	+12%	5.11 (bottom)
Dancers (skirt)	0.919	0.940	+26%	5.12 (bottom)
Family (raised arm)	0.879	0.880	+1%	5.13 (bottom)

Table 5.1: Structural similarity index [WBSS04] between the reference frame at t_1 and the images warped to t_1 either without (auto) or with (user) correction. While both scores are already very good, user interaction yields a further reduction of the remaining error (RRE) of up to 26%.

User interaction for the Capoeira and Dancers scenes removes 12–26% of the remaining error. In contrast, the Family scene shows only a 1% reduction, although the raised-arm artifact is the visually most obvious and disturbing of all examples. The close scores are probably caused by the color similarity of arm and background.

Summative Evaluation. Since there are no scene flow editing approaches to compare against, the relative attractiveness of the presented approach was evaluated against the industry standard, fixing rendered frames with image-space tools [Sey08]. The presented tools are meant for trained visual artists and not for the general public. As such, a summative evaluation was undertaken with 4 experts in the age range 25–35 with at least 5 years of image processing experience as well as exposure to stereo and/or optical flow. They were coached in the use of the tools for up to 20 minutes each, on training footage not used for subsequent evaluation, the latter of which used the Capoeira

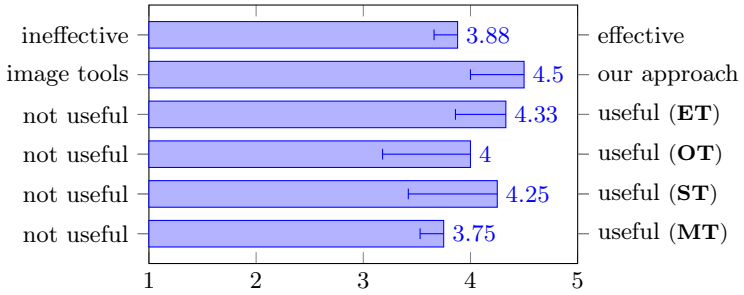


Figure 5.17: User study with scores 1..5; mean values and standard deviation are shown. Sample size was 4 experts with both image editing and stereo and/or optical flow experience. Questions were general usefulness, relative usefulness of the presented tools compared to image-based tools, and usefulness per tool.

scene, Figures 5.11 and 5.14. Assuming a moderate number of 10 frames rendered from an automatically estimated scene flow at $t_{0.0}$, $t_{0.1}$, $t_{0.2}$, .., $t_{1.0}$, The experts were first asked to repair the scene flow for subsequent re-rendering. For the alternative workflow, they were asked to repair the 10 originally rendered frames instead, using an image-space tool of their choice (all chose Adobe Photoshop^{TM}). At 10 minutes into either task, the experts were instructed to finish their last operation. Afterwards, the results of both workflows were compared and the experts were asked to rate both with 1 (not useful) to 5 (very useful) on the MOS (mean opinion score) scale.

As shown in Figure 5.17, the general usefulness of the presented approach for the given task was confirmed with a mean score of 3.88. All experts expressed the wish for real-time scene flow re-computation after each scribble, which is not yet possible with state-of-the-art algorithms. Single wishes included using scribbles on target frames instead of the source frame, and instant comparison against the effect of previous scribbles. Compared to image-space tools, the relative attractiveness of the presented method increased to a mean score of 4.50, with all experts seeing the built-in spatiotemporal consistency between frames as a key advantage of the presented approach.

The experts found all four tools similarly useful with mean scores ranging from 3.75–4.33; all noted that 20 minutes of coaching were sufficient to use the tools effectively. Given the fact that they are used to Photoshop$^{\text{TM}}$, all experts noted that an increasing familiarity with the presented tools would probably allow for even better results.

5.7 Discussion

All experts agreed on the effectiveness of the presented approach as demonstrated by the improved visual quality of the output frames. The most desired improvement were instant response times, which requires real-time scene flow algorithms. With respect to the latter, the scene flow optimization could potentially benefit from a primal-dual approach in the style of [ZPB07], left for future work.

The employed scene flow estimation algorithm is a GPU-based re-implementation of [BMK13]. By integrating user guidance, similar gains can be expected for other scene flow approaches since the failure modes are based on commmon assumptions and scene properties rather than algorithmic intricacies. Should a method without failure cases emerge, the presented tools would become unnecessary. However, this case is not exceedingly likely.

When used for visual media production, the new method does not preclude the use of image-based tools; it optionally preceeds it, reducing but in some cases not fully mitigating image-space work.

The approach has two time savers: First the approximate way most scribbles can be defined (edges between two salient regions being the sole exception), and second the arbitrary number of frames that can be rendered from a single scene flow field. Additionally, when going from traditional media production towards free spacetime navigation, "post-production frame correction" is not possible because the number of frames is arbitrary. In this case, editing depth and motion itself is the only viable way to produce artifact-free output frames.

6 Summary

In my dissertation, I have demonstrated through three examples that the combination of human scene recognition with algorithmic refinement yields powerful, flexible tools well suited for high-quality spatiotemporal reconstruction and computer graphics production while requiring considerably reduced user interaction efforts. My work has shown that correspondence field editing both within and for repeated optimization is a viable and in many cases preferable alternative to output image editing, particularly if many output frames are rendered from few correspondence fields.

In spatial reconstruction, I established how restricting cost volumes or, more generally, the solution space of any energy formulation, by employing user-defined cost blocks can remove a large class of errors while still allowing precise algorithmic solutions.

In temporal reconstruction, I demonstrated how approximate user-given displacements that are wrong by several pixels can be snapped to the subpixel-precise correct location using algorithmic refinement on finer pyramid levels coupled with anisotropic regularization; and

how locally adapted smoothness weights compensate for structure and texture changes within a recorded scene.

In full spatiotemporal reconstruction, I presented a combination of four tools to enable better solutions: discontinuities being encouraged by scribbling edges; occlusions and color constancy violations being marked to eliminate outliers; and the approximate displacements and local smoothness adaptations of temporal reconstruction also holding in the spatial and spatiotemporal domain.

Correspondence field editing is a relatively new area of research. My work on stereo editing has contributed new ideas to the small but growing pool of tools suitable for stereoscopic visual media productions. My research on optical flow editing stands relatively exclusive, with few high-quality commercial tools[1] and one approach from our own workgroup [KRLM11] as lone contemporaries. I was, to the best of my knowledge, the first to explore scene flow editing, an emerging field that becomes slowly more attractive as substituting stereo and optical flow by scene flow, long desired but algorithmically intricate and computationally expensive, finally approaches becoming feasible [VSR13].

Due to the need for interactive performance, the runtime of the underlying optimizations has been the limiting factor for researching ever more or better scene-oriented tools. In particular, occlusion handling using symmetric estimation doubles or, in spatiotemporal

[1] RE:Vision Twixtor: `http://www.revisionfx.com/products/twixtor/`

reconstruction, multiplies runtimes linearly. The modeling of more involved energy formulations is a second frontier for further intuitive user interactions; most notably, mixed pixels whose handling is standard in the matting community have yet to appear in widespread form within correspondence estimation approaches.

With faster and more comprehensively modeled algorithms in the field of spatiotemporal reconstruction expected in the future, I am curious and excited to see my tools being extended and/or succeeded by emerging possibilities and yet-to-be-developed forms of higher-level algorithmic user interaction that will help shape tomorrow's standard of realistic computer graphics.

Bibliography

[ADPS07] L. Alvarez, R. Deriche, T. Papadopoulo, and J. Sánchez. „Symmetrical dense optical flow estimation with occlusions detection“. In: *International Journal of Computer Vision* 75.3 (2007), pp. 371–385.

[BBM+01] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. „Unstructured lumigraph rendering“. In: *Proc. of 28th SIGGRAPH*. ACM. 2001, pp. 425–432.

[BBM09] T. Brox, C. Bregler, and J. Malik. „Large displacement optical flow“. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 0 (2009), pp. 41–48. DOI: `http://doi.ieeecomputersociety.org/10.1109/CVPRW.2009.5206697`.

[BBPW04] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. „High accuracy optical flow estimation based on a theory for warping“. In: *Computer Vision-ECCV 2004*. Springer, 2004, pp. 25–36.

[BMK13] T. Basha, Y. Moses, and N. Kiryati. „Multi-view scene flow estimation: A view centered variational approach“. In: *International journal of computer vision* 101.1 (2013), pp. 6–21.

[BN92] T. Beier and S. Neely. „Feature-based image metamorphosis“. In: *Proc. 19th Computer Graphics and Interactive Techniques*. SIGGRAPH. 1992, pp. 35–42.

[BRA+11] K. Berger, K. Ruhl, M. Albers, Y. Schröder, A. Scholz, S. Guthe, and M. Magnor. „The capturing of turbulent gas flows using multiple Kinects“. In: *Proc. CDC4CV 2011*. ISBN: 978-1-4673-0061-2. IEEE. Nov. 2011.

[BRB+11] K. Berger, K. Ruhl, C. Brümmer, Y. Schröder, A. Scholz, and M. Magnor. „Markerless Motion Capture using multiple Color-Depth Sensors“. In: *Proc. Vision, Modeling and Visualization (VMV) 2011*. Oct. 2011, pp. 317–324.

[BSL+07] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. „A Database and Evaluation Methodology for Optical Flow“. In: *IEEE International Conference on Computer Vision*. 2007, pp. 1–8. DOI: http://doi.ieeecomputersociety.org/10.1109/ICCV.2007.4408903.

[BSL+11] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. „A database and evaluation methodology

for optical flow“. In: *International Journal of Computer Vision* 92.1 (2011), pp. 1–31.

[BWSB12] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. „A naturalistic open source movie for optical flow evaluation“. In: *European Conf. on Computer Vision (ECCV)*. Ed. by A. Fitzgibbon et al. (Eds.) Part IV, LNCS 7577. Springer-Verlag, Oct. 2012, pp. 611–625.

[BYJ14] L. Bao, Q. Yang, and H. Jin. „Fast Edge-Preserving PatchMatch for Large Displacement Optical Flow“. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2014, pp. 3534–3541.

[Cha04] A. Chambolle. „An Algorithm for Total Variation Minimization and Applications“. In: *Journal of Mathematical Imaging and Vision* 20.1-2 (2004), pp. 89–97. DOI: 10.1023/B:JMIV.0000011325.36760.1e. URL: http://dx.doi.org/10.1023/B:JMIV.0000011325.36760.1e.

[Cro84] F. C. Crow. „Summed-area tables for texture mapping“. In: *Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '84. New York, NY, USA: ACM, 1984, pp. 207–212. ISBN: 0-89791-138-5. DOI: 10.1145/800031.808600. URL: http://doi.acm.org/10.1145/800031.808600.

[CSD11] G. Chaurasia, O. Sorkine, and G. Drettakis. „Silhouette-aware warping for image-based rendering“. In: *Proceedings of the 22nd Eurographics conference on Rendering*. 2011, pp. 1223–1232.

[CSH11] J. Cech, J. Sanchez-Riera, and R. Horaud. „Scene flow estimation by growing correspondence seeds“. In: *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*. 2011, pp. 3129–3136. DOI: 10.1109/CVPR.2011.5995442. URL: http://dx.doi.org/10.1109/CVPR.2011.5995442.

[Dab10] K. Dabov. *Image and video restoration with nonlocal transform-domain filtering*. PhD thesis. Tampere University of Technology, 2010. ISBN: 978-952-15-2421-9.

[DCSK14] Y. Doron, N. D. Campbell, J. Starck, and J. Kautz. „User Directed Multi-View-Stereo“. In: *2nd Workshop on User-Centred Computer Vision (at ACCV)*. 2014.

[DYLT05] Y. Deng, Q. Yang, X. Lin, and X. Tang. „A Symmetric Patch-Based Correspondence Model for Occlusion Handling“. In: *10th IEEE International Conference on Computer Vision (ICCV 2005), 17-20 October 2005, Beijing, China*. 2005, pp. 1316–1322. DOI: 10.1109/ICCV.2005.

23. URL: http://doi.ieeecomputersociety.org/10.1109/ICCV.2005.23.

[ENM11] R. D. Eastman, N. S. Netanyahu, and J. L. Moigne. „Survey of image registration methods“. In: *Image Registration for Remote Sensing*. Cambridge University Press, 2011, pp. 35–78.

[Fai13] I. Failes. *Creature features: VFX in TVCs*. 2013. URL: http://www.fxguide.com/featured/creature-features-vfx-in-tvcs/.

[Fai14] I. Failes. *Divergent: making the mirror room*. 2014. URL: http://www.fxguide.com/featured/divergent-making-the-mirror-room/.

[FH06] P. Felzenszwalb and D. Huttenlocher. „Efficient Belief Propagation for Early Vision“. In: *International Journal of Computer Vision*. Vol. 70. 1. Springer, 2006, pp. 41–54.

[Fre14] V. Frei. *Guardians of the Galaxy: Pete Travers Interview*. 2014. URL: http://www.artofvfx.com/?p=8894.

[GLU12] A. Geiger, P. Lenz, and R. Urtasun. „Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite“. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[GWCO09] M. Guttmann, L. Wolf, and D. Cohen-Or. „Semi-automatic stereo extraction from video footage.“ In: *Intl. Conference on Computer Vision (ICCV)*. 2009, pp. 136–142.

[HB11] S. Hadfield and R. Bowden. „Kinecting the dots: Particle based scene flow from depth sensors“. In: *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*. 2011, pp. 2290–2295. DOI: 10.1109/ICCV.2011.6126509. URL: http://dx.doi.org/10.1109/ICCV.2011.6126509.

[HBR+11] A. Hosni, M. Bleyer, C. Rhemann, M. Gelautz, and C. Rother. „Real-time local stereo matching using guided image filtering“. In: *Multimedia and Expo (ICME), 2011 IEEE International Conference on*. IEEE. 2011, pp. 1–6.

[HD07] F. Huguet and F. Devernay. „A Variational Method for Scene Flow Estimation from Stereo Sequences“. In: *IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007*. 2007, pp. 1–7. DOI: 10.1109/ICCV.2007.4409000. URL: http://dx.doi.org/10.1109/ICCV.2007.4409000.

[Her09] G. T. Herman. *Fundamentals of Computerized Tomography: Image Reconstruction from Projections*. Advances in Pattern Recognition. Springer, 2009. ISBN: 978-1-85233-

617-2. DOI: 10.1007/978-1-84628-723-7. URL: http://dx.doi.org/10.1007/978-1-84628-723-7.

[HLKK14] Y. Hwang, J. Lee, I. Kweon, and S. J. Kim. „Color Transfer Using Probabilistic Moving Least Squares“. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*. 2014, pp. 3342–3349. DOI: 10.1109/CVPR.2014.427. URL: http://dx.doi.org/10.1109/CVPR.2014.427.

[HRB+13] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. „Fast Cost-Volume Filtering for Visual Correspondence and Beyond“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 35.2 (2013), pp. 504 –511.

[HRT+09] N. Hasler, B. Rosenhahn, T. Thormahlen, M. Wand, J. Gall, and H. Seidel. „Markerless motion capture with unsynchronized moving cameras“. In: *Computer Vision and Pattern Recognition, 2009*. IEEE. 2009, pp. 224–231.

[HST10] K. He, J. Sun, and X. Tang. „Guided image filtering“. In: *Computer Vision–ECCV 2010*. Springer, 2010, pp. 1–14.

[IM06] M. Isard and J. MacCormick. „Dense Motion and Disparity Estimation Via Loopy Belief Propagation“. In: *Computer Vision - ACCV 2006, 7th Asian Conference*

on Computer Vision, Hyderabad, India, January 13-16, 2006, Proceedings, Part II. 2006, pp. 32–41. DOI: 10.1007/11612704_4. URL: http://dx.doi.org/10.1007/11612704_4.

[Kas92] M. Kass. „Inverse problems in computer graphics“. In: *Creating and animating the virtual world*. Springer, 1992, pp. 21–33.

[KBKL09] A. Kolb, E. Barth, R. Koch, and R. Larsen. „Time-of-flight sensors in computer graphics“. In: *Eurographics State of the Art Reports* (2009), pp. 119–134.

[KLM10] F. Klose, C. Lipski, and M. Magnor. „Reconstructing Shape and Motion from Asynchronous Cameras“. In: *Proc. Vision, Modeling and Visualization (VMV) 2010*. Siegen, Germany, Nov. 2010, pp. 171–177.

[KLR+11] F. Klose, C. Lipski, K. Ruhl, B. Meyer, and M. Magnor. „A Toolchain for Capturing and Rendering Stereo and Multi-View Datasets“. In: *Proc. The International Conference on 3D Imaging (IC3D) 2011*. Dec. 2011, pp. 1–7.

[Kol06] V. Kolmogorov. „Convergent tree-reweighted message passing for energy minimization“. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 28.10 (2006), pp. 1568–1583.

[KPZ+11] C. Kuster, T. Popa, C. Zach, C. Gotsman, M. Gross, P. Eisert, J. Hornegger, and K. Polthier. „FreeCam: A Hybrid Camera System for Interactive Free-Viewpoint Video“. In: *Vision, Modeling, and Visualization (VMV)*. 2011, pp. 17–24.

[KRL+11] F. Klose, K. Ruhl, C. Lipski, C. Linz, and M. Magnor. „Stereoscopic 3D view synthesis from unsynchronized multi-view video“. In: *Proc. European Signal Processing Conference (EUSIPCO) 2011*. Ed. by F. Klose, K. Ruhl, C. Lipski, C. Linz, and M. Magnor. Barcelona, Spain, May 2011, pp. 1904–1909.

[KRLM11] F. Klose, K. Ruhl, C. Lipski, and M. Magnor. „Flowlab - an interactive tool for editing dense image correspondences“. In: *Proc. European Conference on Visual Media Production (CVMP) 2011*. Aug. 2011, pp. 1–8.

[KTS09] S. Kosov, T. Thormählen, and H.-P. Seidel. „Accurate real-time disparity estimation with variational methods“. In: *Advances in Visual Computing*. Springer, 2009, pp. 796–807.

[KZ04] V. Kolmogorov and R. Zabin. „What energy functions can be minimized via graph cuts?“ In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 26.2 (2004), pp. 147–159.

[LK11] W. Lin and C. J. Kuo. „Perceptual visual quality metrics: A survey“. In: *J. Visual Communication and Image Representation* 22.4 (2011), pp. 297–312. DOI: 10.1016/j.jvcir.2011.01.005. URL: http://dx.doi.org/10.1016/j.jvcir.2011.01.005.

[LKRM11a] C. Lipski, F. Klose, K. Ruhl, and M. Magnor. „Making of Who Cares HD Stereoscopic Free Viewpoint Video“. In: *Proc. European Conference on Visual Media Production (CVMP) 2011*. Vol. 8. Nov. 2011, pp. 1–10.

[LKRM11b] C. Lipski, F. Klose, K. Ruhl, and M. Magnor. „The virtual video camera: Simplified 3DTV acquisition and processing“. In: *Proc. 3DTV-CON 2011*. Antalya: IEEE Computer Society, May 2011, pp. 1–4.

[LLM10] C. Linz, C. Lipski, and M. Magnor. „Multi-Image Interpolation based on Graph-Cuts and Symmetric Optical Flow“. In: *Proc. Vision, Modeling and Visualization (VMV) 2010*. Eurographics. Siegen, Germany: Eurographics Association, Nov. 2010, pp. 115–122.

[LLN+12] C. Lipski, C. Linz, T. Neumann, M. Wacker, and M. Magnor. „High Resolution Image Correspondences for Video Post-Production“. In: *Journal of Virtual Reality and Broadcasting (JVRB)* 9.2012.8 (Dec. 2012), pp. 1–12.

[LLR+10] C. Linz, C. Lipski, L. Rogge, C. Theobalt, and M. Magnor. „Space-Time Visual Effects as a Post-Production Process“. In: *ACM Multimedia 2010 - 1st Intl. 3DVP Workshop*. 2010.

[LLW08] A. Levin, D. Lischinski, and Y. Weiss. „A Closed-Form Solution to Natural Image Matting“. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 30.2 (2008), pp. 228–242. DOI: 10.1109/TPAMI.2007.1177. URL: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2007.1177.

[Lof12] M. Loftus. *VFX for TV Series*. 2012. URL: http://www.postmagazine.com/Publications/Post-Magazine/2012/April-1-2012/VFX-for-TV-Series.

[Low99] D. G. Lowe. „Object Recognition from Local Scale-Invariant Features“. In: *Computer Vision (ICCV), 1999 International Conference on*. 1999, pp. 1150–1157. URL: http://computer.org/proceedings/iccv/0164/vol%202/01641150abs.htm.

[LYT11] C. Liu, J. Yuen, and A. Torralba. „Sift flow: Dense correspondence across scenes and its applications“. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 33.5 (2011), pp. 978–994.

[Mor12] P. Mordohai. „On the evaluation of scene flow estimation“. In: *Computer Vision – ECCV 2012. Workshops and Demonstrations*. Springer. 2012, pp. 148–157.

[MSZ+11] X. Mei, X. Sun, M. Zhou, H. Wang, X. Zhang, et al. „On building an accurate stereo matching system on graphics hardware“. In: *GPU Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE. 2011, pp. 467–474.

[MW15] M. G. Mozerov and J. van de Weijer. „Accurate Stereo Matching by Two-Step Energy Minimization“. In: *IEEE Transactions on Image Processing* 24.3 (2015), pp. 1153–1163. DOI: 10.1109/TIP.2015.2395820. URL: http://dx.doi.org/10.1109/TIP.2015.2395820.

[NIH+11] R. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. „KinectFusion: Real-time dense surface mapping and tracking“. In: *10th Int. Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE. 2011, pp. 127–136.

[NLD11] R. Newcombe, S. Lovegrove, and A. Davison. „DTAM: Dense tracking and mapping in real-time“. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2320–2327.

[NML+13] R. Nair, S. Meister, M. Lambers, M. Balda, H. G. Hofmann, A. Kolb, D. Kondermann, and B. Jähne. „Ground Truth for Evaluating Time of Flight Imaging“. In: *Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications - Dagstuhl 2012 Seminar on Time-of-Flight Imaging and GCPR 2013 Workshop on Imaging New Modalities*. 2013, pp. 52–74. DOI: 10.1007/978-3-642-44964-2_4. URL: http://dx.doi.org/10.1007/978-3-642-44964-2_4.

[PBH12] F. Pitié, G. Baugh, and J. Helms. „DepthArtist: a stereoscopic 3D conversion tool for CG animation“. In: *Proceedings of the 9th European Conference on Visual Media Production*. 2012, pp. 32–39.

[PTK89] T. Poggio, V. Torre, and C. Koch. „Computational vision and regularization theory“. In: *Image understanding* 3.1-18 (1989), p. 111.

[QDC13] J. Quiroga, F. Devernay, and J. L. Crowley. „Local/global scene flow estimation“. In: *ICIP-IEEE International Conference on Image Processing*. IEEE. 2013.

[REH+15] K. Ruhl, M. Eisemann, A. Hilsmann, P. Eisert, and M. Magnor. „Interactive Scene Flow Editing for Improved Image-based Rendering and Virtual Spacetime Navigation“. In: *Currently under review*. 2015.

[REM13] K. Ruhl, M. Eisemann, and M. Magnor. „Cost Volume-based Interactive Depth Editing in Stereo Post-processing“. In: *Euro. Conference on Visual Media Production (CVMP)*. Vol. 10. Nov. 2013, pp. 1–6.

[RGPB12] R. Ranftl, S. Gehrig, T. Pock, and H. Bischof. „Pushing the limits of stereo using variational stereo estimation“. In: *2012 IEEE Intelligent Vehicles Symposium, IV 2012, Alcal de Henares, Madrid, Spain, June 3-7, 2012*. 2012, pp. 401–407. DOI: 10.1109/IVS.2012.6232171. URL: http://dx.doi.org/10.1109/IVS.2012.6232171.

[RHB+11] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz. „Fast cost-volume filtering for visual correspondence and beyond“. In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE. 2011, pp. 3017–3024.

[RHK+12] K. Ruhl, B. Hell, F. Klose, C. Lipski, S. Petersen, and M. Magnor. „Improving Dense Image Correspondence Estimation with Interactive User Guidance“. In: *Proc. ACM Multimedia 2012*. ACM, Oct. 2012, pp. 1129–1132.

[RK09] D. Ring and A. Kokaram. „User-Assisted Feature Correspondence Matching“. In: *Proceedings of the 2009 Conference for Visual Media Production*. 2009, pp. 214–219.

[RKLM12] K. Ruhl, F. Klose, C. Lipski, and M. Magnor. „Integrating Approximate Depth Data into Dense Image Correspondence Estimation“. In: *Proc. European Conference on Visual Media Production (CVMP) 2012*. Vol. 9. Dec. 2012, pp. 1–6.

[ROS10] ROS. *Kinect calibration guide*. `http://www.ros.org/wiki/kinect_calibration/technical`. 2010.

[RRKB11] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. „ORB: an efficient alternative to SIFT or SURF“. In: *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2564–2571.

[RWF+13] K. Ruhl, S. Wenger, D. Franke, J. Saretzki, and M. Magnor. *Fine-Scale Editing of Continuous Volumes using Adaptive Surfaces*. Poster at VMV. Sept. 2013.

[SAB02] J. Salvi, X. Armangué, and J. Batlle. „A comparative review of camera calibrating methods with accuracy evaluation“. In: *Pattern Recognition* 35.7 (2002), pp. 1617 –1635. ISSN: 0031-3203. DOI: `http://dx.doi.org/10.1016/S0031-3203(01)00126-1`. URL: `http://www.sciencedirect.com/science/article/pii/S0031320301001261`.

[SCD+06] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. „A comparison and evaluation of multi-view

stereo reconstruction algorithms“. In: *Computer Vision and Pattern Recognition, IEEE* 1 (2006), pp. 519–528.

[Sey06] M. Seymour. „Art of Optical Flow“. In: *FXguide* (2006). URL: `http://www.fxguide.com/featured/art_of_optical_flow/`.

[Sey08] M. Seymour. „Art of Digital 3D Stereoscopic Film“. In: *FXguide* (2008). URL: `http://www.fxguide.com/featured/art_of_digital_3d_stereoscopic_film/`.

[Sey12a] M. Seymour. *Art of Stereo Conversion: 2D to 3D*. 2012. URL: `http://www.fxguide.com/featured/art-of-stereo-conversion-2d-to-3d-2012/`.

[Sey12b] M. Seymour. „Pixel Farm's PFDepth: exclusive first look“. In: *FXguide* (2012). URL: `http://www.fxguide.com/featured/pixel-farms-pfdepth-exclusive-first-look/`.

[Sey12c] M. Seymour. *VFX roll call for The Avengers*. 2012. URL: `http://www.fxguide.com/featured/vfx-roll-call-for-the-avengers/`.

[SFC+11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. „Real-time human pose recognition in parts from single depth im-

ages“. In: *Computer Vision and Pattern Recognition*. Vol. 2. 2011, p. 7.

[SLK05] J. Sun, Y. Li, and S. B. Kang. „Symmetric Stereo Matching for Occlusion Handling“. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*. 2005, pp. 399–406. DOI: 10.1109/CVPR.2005.337. URL: http://dx.doi.org/10.1109/CVPR.2005.337.

[SLP14] D. Sun, C. Liu, and H. Pfister. „Local layering for joint motion estimation and occlusion detection“. In: *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE. 2014, pp. 1098–1105.

[SPC09] F. Steinbruecker, T. Pock, and D. Cremers. „Large Displacement Optical Flow Computation without Warping“. In: *ICCV*. 2009, pp. 1609–1614.

[SPH+11] A. Smolic, S. Poulakos, S. Heinzle, P. Greisen, M. Lang, A. Hornung, M. Farre, N. Stefanoski, O. Wang, L. Schnyder, R. Monroy, and M. Gross. „Disparity-Aware Stereo 3D Production Tools“. In: *Proceedings of the 2011 Conference for Visual Media Production*. 2011, pp. 165–173.

[SRB10] D. Sun, S. Roth, and M. J. Black. „Secrets of optical flow estimation and their principles“. In: *Computer Vision*

and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE. 2010, pp. 2432–2439.

[SRM12] A. Sellent, K. Ruhl, and M. Magnor. „A Loop-Consistency Measure for Dense Correspondences in Multi-View Video“. In: *Journal of Image and Vision Computing* 30.9 (June 2012), pp. 641–654.

[SS02] D. Scharstein and R. Szeliski. „A taxonomy and evaluation of dense two-frame stereo correspondence algorithms“. In: *International Journal of Computer Vision* 47.1-3 (2002), pp. 7–42.

[SSB10] D. Sun, E. Sudderth, and M. Black. „Layered Image Motion with Explicit Occlusions, Temporal Consistency, and Depth Ordering“. In: *Advances in Neural Information Processing Systems* (2010), pp. 2226–2234.

[SSJ+10] D. Sýkora, D. Sedlacek, S. Jinchao, J. Dingliana, and S. Collins. „Adding Depth to Cartoons Using Sparse Depth (In)equalities“. In: *Computer Graphics Forum* 29.2 (2010), pp. 615–623.

[SSS06] N. Snavely, S. Seitz, and R. Szeliski. „Photo tourism: exploring photo collections in 3D“. In: *ACM Transactions on Graphics (TOG)*. Vol. 25. 3. ACM. 2006, pp. 835–846.

[SSS14] S. N. Sinha, D. Scharstein, and R. Szeliski. „Efficient High-Resolution Stereo Matching Using Local Plane Sweeps“. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*. 2014, pp. 1582–1589. DOI: 10.1109/CVPR.2014.205. URL: http://dx.doi.org/10.1109/CVPR.2014.205.

[STDT08] S. Schuon, C. Theobalt, J. Davis, and S. Thrun. „High-quality scanning using time-of-flight depth superresolution“. In: *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08.* IEEE. 2008, pp. 1–7.

[Ste13a] A. Steadman. *The Glory of Twixtor: Unorthodox Uses of Faux Slow-Mo*. 2013. URL: http://petapixel.com/2013/06/11/the-glory-of-twixtor-unorthodox-uses-of-faux-slow-mo/.

[Ste13b] J. Steurer. „Tri-Focal Rig (Practical Camera Configurations for Image and Depth Acquisition)“. In: *SMPTE Conferences* 2013.10 (2013), pp. 1–15.

[TSF12] E. Tola, C. Strecha, and P. Fua. „Efficient large-scale multi-view stereo for ultra high-resolution image sets“. In: *Machine Vision and Applications* 23.5 (Sept. 2012), pp. 903–920. ISSN: 0932-8092. DOI: 10.1007/s00138-

011-0346-8. URL: http://dx.doi.org/10.1007/s00138-011-0346-8.

[VBR+99] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. „Three-dimensional scene flow“. In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 2. IEEE. 1999, pp. 722–729.

[VBZ+10] L. Valgaerts, A. Bruhn, H. Zimmer, J. Weickert, C. Stoll, and C. Theobalt. „Joint estimation of motion, structure and geometry from stereo sequences“. In: *Computer Vision–ECCV 2010*. Springer, 2010, pp. 568–581.

[VSR11] C. Vogel, K. Schindler, and S. Roth. „3D scene flow estimation with a rigid motion prior“. In: *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*. 2011, pp. 1291–1298. DOI: 10.1109/ICCV.2011.6126381. URL: http://dx.doi.org/10.1109/ICCV.2011.6126381.

[VSR13] C. Vogel, K. Schindler, and S. Roth. „Piecewise Rigid Scene Flow“. In: *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*. 2013, pp. 1377–1384. DOI: 10.1109/ICCV.2013.174. URL: http://dx.doi.org/10.1109/ICCV.2013.174.

[WAC07] J. Wang, M. Agrawala, and M. F. Cohen. „Soft scissors: an interactive tool for realtime high quality matting“. In: *ACM Transactions on Graphics (TOG)* 26.3 (2007), 9:1–9:6. DOI: 10.1145/1276377.1276389. URL: http://doi.acm.org/10.1145/1276377.1276389.

[WACS11] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. „Multicore bundle adjustment“. In: *The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011*. 2011, pp. 3057–3064. DOI: 10.1109/CVPR.2011.5995552. URL: http://ccwu.me/vsfm/.

[WBSS04] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli. „Image quality assessment: from error visibility to structural similarity“. In: *Image Processing, IEEE Transactions on* 13.4 (2004), pp. 600–612. ISSN: 1057-7149. DOI: 10.1109/TIP.2003.819861.

[WBV+11] A. Wedel, T. Brox, T. Vaudrey, C. Rabe, U. Franke, and D. Cremers. „Stereoscopic Scene Flow Computation for 3D Motion Understanding“. In: *International Journal of Computer Vision* 95.1 (2011), pp. 29–51.

[WC07] J. Wang and M. F. Cohen. „Image and Video Matting: A Survey“. In: *Foundations and Trends in Computer Graphics and Vision* 3.2 (2007), pp. 97–175. DOI: 10.

1561/0600000019. URL: http://dx.doi.org/10.1561/0600000019.

[Wil09] L. Wilkes. „The Role of Ocula in Stereo Post Production“. In: *The Foundry, Whitepaper* (2009). URL: http://www.thefoundry.co.uk/.

[WJYG08] L. Wang, H. Jin, R. Yang, and M. Gong. „Stereoscopic inpainting: Joint color and depth completion from stereo images“. In: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*. 2008. DOI: 10.1109/CVPR.2008.4587704. URL: http://dx.doi.org/10.1109/CVPR.2008.4587704.

[WLF+11] O. Wang, M. Lang, M. Frei, A. Hornung, A. Smolic, and M. Gross. „StereoBrush: interactive 2D to 3D conversion using discontinuous warps“. In: *Proceedings of the Eigth Eurographics Symposium on Sketch-Based Interfaces and Modeling*. 2011, pp. 47–54.

[Wol98] G. Wolberg. „Image morphing: a survey“. In: *The Visual Computer* 14.8/9 (1998), pp. 360–372. DOI: 10.1007/s003710050148. URL: http://dx.doi.org/10.1007/s003710050148.

[WTP+09] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. „Anisotropic Huber-L1 Optical

Flow“. In: *Proceedings of the British Machine Vision Conference (BMVC)*. Vol. 34. London, UK, 2009, pp. 1–11.

[WY11] L. Wang and R. Yang. „Global stereo matching leveraged by sparse ground control points“. In: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*. 2011, pp. 3033–3040.

[ZKU+04] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. A. J. Winder, and R. Szeliski. „High-quality video view interpolation using a layered representation“. In: *ACM Trans. Graph.* 23.3 (2004), pp. 600–608.

[ZPB07] C. Zach, T. Pock, and H. Bischof. „A Duality Based Approach for Realtime TV-L1 Optical Flow“. In: *Pattern recognition: 29th DAGM symposium*. Vol. 29. 2007, pp. 214–223.

[ZPCY13] C. Zhang, B. Price, S. Cohen, and R. Yang. „High-Quality Stereo Video Matching via User Interaction and Space-Time Propagation“. In: *Intl. Conf. 3DV*. IEEE. 2013, pp. 71–78.

[Mic10] Microsoft Corporation. *Kinect for Xbox 360*. Redmond WA. 2010. URL: `http://news.microsoft.com/2010/06/13/kinect-for-xbox-360-is-official-name-of-microsofts-controller-free-game-device/`.

Glossary

C_k	camera at index k.
∂	partial derivative, subgradient.
∇	sum of partial derivatives.
i	iteration step, starting at 0.
I_t^k	image from camera with index k at time t.
k	camera index; hero camera generally $k = 0$.
$\mathbf{K}$	intrinsic camera calibration matrix.
$\mathbf{p}_t^k$	discrete or continuous 2D pixel coordinate, re-projected to camera k at time t.
π	projection between cameras or between world and image space, using $\mathbf{K}$ and $\mathbf{S}$.
$\mathbf{Q}$	solution of a spatiotemporal reconstruction.
$\mathbf{Q}_{\mathrm{st}}$	solution for stereo, with $\mathbf{Q}_{\mathrm{st}} = [z]$.
$\mathbf{Q}_{\mathrm{of}}$	solution for optical flow, with $\mathbf{Q}_{\mathrm{of}} = [u, v]^T$.
$\mathbf{Q}_{\mathrm{sf}}$	solution for scene flow, with $\mathbf{Q}_{\mathrm{sf}} = [z, U, V, W]^T$.
$\mathbf{S}$	extrinsic camera calibration matrix.
t	time, frame index starting at 0.

u	horizontal motion, $-u_{max}..+u_{max}$, in 2D image space.
U	horizontal motion, $-U_{max}..+U_{max}$, in 3D world space.
v	vertical motion, $-v_{max}.. + v_{max}$, in 2D image space.
V	vertical motion, $-V_{max}.. + V_{max}$, in 3D world space.
W	z-motion, $-W_{max}.. + W_{max}$, in 3D world space.
$\mathbf{x}$	discrete 2D pixel coordinate in the hero camera.
z	depth value, $0..z_{max}$, same in image and world space.